

École Polytechnique de l'Université de Tours
64, Avenue Jean Portalis
37200 TOURS, FRANCE
Tél. +33 (0)2 47 36 14 14
www.polytech.univ-tours.fr

Département Informatique
5e année
2011-2012

Rapport de projet de fin d'études

Recalage 2D-3D en dermatologie

Le 8 mai 2012

Etudiant :

Alexandre NEVEU
<alexandre.neveu@etu.univ-tours.fr>

Encadrants :

Pascal MAKRIS
<pascal.makris@univ-tours.fr>
Jean-Yves RAMEL
<jean-yves.ramel@univ-tours.fr>
Baudouin MARTIN
<baudouin.martin@univ-tours.fr>

TABLE DES MATIERES



REMERCIEMENTS

Je tiens sincèrement à remercier l'ensemble de mes encadrants, Mr Pascal MAKRIS, Mr Jean-Yves RAMEL, pour leur patience, leur disponibilité et leur soutien tout au long de ce projet, et surtout pendant les périodes compliquées, notamment lors des deux mois de modélisation et de conception de l'application. Remerciements tout particuliers à Mr Baudouin MARTIN qui a consacré beaucoup de son temps à mes côtés pour réaliser la modélisation de l'application.

Je tiens également à remercier le client, Jean-Marc GREGOIRE, de l'équipe 5 de l'UMRS Inserm U930, qui a également été un soutien très important tout au long de l'année et qui s'est toujours montré disponible et investi dans le projet pour m'aider à le mener à bien. Encore merci pour votre patience et votre investissement dans ce projet sans lesquels je n'aurais probablement pas été aussi loin.



INTRODUCTION

Dans le cadre de mon projet de fin d'étude, j'ai réalisé une application médicale dont le résultat permettra à un praticien de mieux appréhender la forme réelle d'un mélanome sous cutanée. En effet, à partir d'une échographie d'un mélanome, contenant des points repères ultrasonores, puis à partir d'une photographie de la peau contenant également des points repères, l'application donne le résultat de la fusion de ces deux images.

La première partie de ce rapport traitera de l'appréhension du sujet et notamment de la reprise d'un module d'une application existante, réalisée par un ancien doctorant de Polytech, Ludovic Paulhac. Ce module existant étant l'affichage de l'acquisition 3D, nous verrons dans ce rapport que cette première étape du projet a implicitement entraîné une importante étape de modélisation.

Et cette partie modélisation représente la seconde partie du rapport, la plus importante du projet en terme de temps et de résultat puisque celle-ci m'a permis de mettre en place un projet structuré et surtout modulable, que quiconque pourra reprendre et faire évoluer dans un futur proche. Nous verrons donc en détail l'évolution de cette modélisation au cours du temps et pour laquelle de nombreuses discussions avec les encadrants et le client ont permis d'arriver à un résultat cohérent.

La dernière étape de ce rapport concerne le développement de l'application, qui a également demandé beaucoup d'échanges avec le client pour que le résultat corresponde exactement à ses attentes. Nous étudierons également les différents modules et interfaces développées.

Ce projet était encadré par Pascal MAKRIS, Jean-Yves RAMEL et Baudouin MARTIN tandis que le client était Jean-Marc GREGOIRE (de l'équipe 5 de l'UMRS Inserm U930).

CONTEXTE DE LA REALISATION

1- Contexte

En dermatologie, l'observation d'une lésion ou d'une tumeur se fait de façon visuelle avec éventuellement des instruments optiques (loupe) qui ne permettent pas d'en évaluer la réelle étendue sous cutanée. L'échographie 3D haute résolution permet une exploration en profondeur du derme et donne la réelle étendue sous cutanée. Pour des lésions nécessitant une exérèse (mélanome, carcinome, …), le chirurgien éprouve une réelle difficulté à repositionner l'image échographique sur la zone où se situe la lésion car il n'existe pas de repères précis et l'échelle échographique est modifiée à l'affichage sur l'écran. Un repositionnement parfait de l'étendue sous cutanée d'une lésion (échographie 3D) avec sa partie visible sur le derme (photographie numérique) permettrait d'avoir une chirurgie plus précise avec un risque de récidives plus faible.

Ce travail propose le développement d'un logiciel qui permettra de recaler, de façon semi-automatique (intervention du praticien pour marquer les points repères échographiques et visuels) des plans « Scan C » échographiques obtenus avec un échographe 3D développé en collaboration avec la Société Atys Médical et l'Université de Tours (UMRS Inserm U930 équipe 5) avec une photographie de la surface de la lésion. Un dispositif pour positionner correctement les repères échographiques et visuels est en cours de développement par l'équipe 5 de l'UMRS Inserm U930 (Jean-Marc GREGOIRE). Des essais pourront être réalisés avec le docteur Lester COWELL, chirurgien de la Melanoma Clinic à Perth, en Australie.

2- Description générale

L'existant :

Trois modules sont déjà implémentés et permettent d'arriver à l'affichage de l'échographie 3D, voici une présentation rapide de ces trois modules :

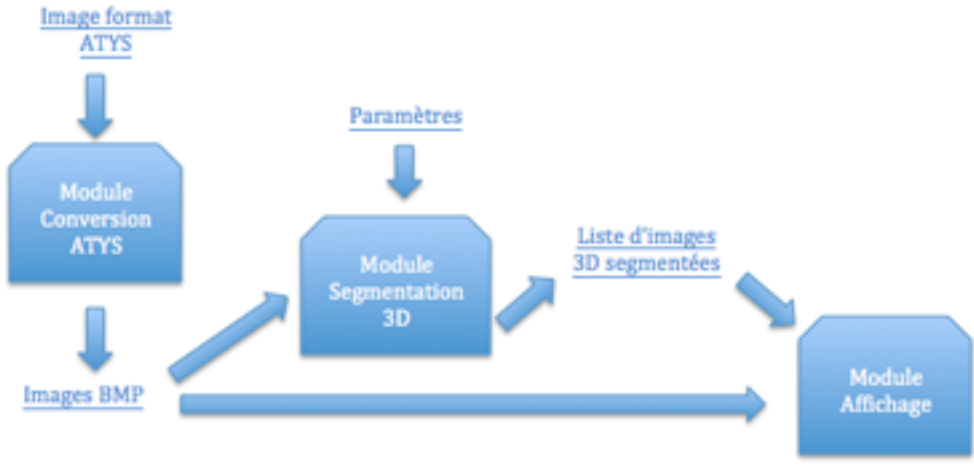


Figure 1 - Modules existants du projet

– Module de conversion (image 3D). Il prend en entrée une image 3D (par exemple de format ATYS, format spécifique à cette entreprise) en une liste d'images BMP.

– Module de segmentation 3D. Développé par Ludovic Paulhac, ce module retourne, à partir des images BMP juste créées par le module précédent, de nouvelles images 3D et segmentées.

– Module d'affichage. A partir des images 3D segmentées ou à partir des images BMP, ce module affiche le résultat obtenu sous forme 3D où l'utilisateur peut faire varier les trois plans pour obtenir la coupe qu'il souhaite et visualiser le rendu. Un aperçu de l'application existante est présenté juste en dessous du schéma d'intégration des nouveaux modules dans le projet existant.

Les modules à intégrer :

– Module affichage. Il faudra modifier ce module de sorte à ce qu'il puisse également gérer l'affichage du résultat du module de fusion ainsi que l'orientation du volume d'acquisition 3D pour afficher des scans C parallèles à la surface de la peau.

– IHM intégrant tous les modules. Cela correspond au second objectif de ce projet, où tous les modules devront être regroupés dans une seule interface afin d'obtenir une application homogène et simple d'utilisation.

– Module de fusion. L'intégration de ce module va nécessiter l'intégration de nouveaux éléments dans le système comme la photo de la peau, les points repères à positionner sur la photo et sur le scan C et également l'alignement des scans C pour qu'ils soient parallèles à la surface de la peau. Une fois que tous ses éléments sont rassemblés, le module peut être lancé et l'affichage du résultat permettra à l'utilisateur de faire varier la transparence et l'affichage des scans C et de la photo de la peau.

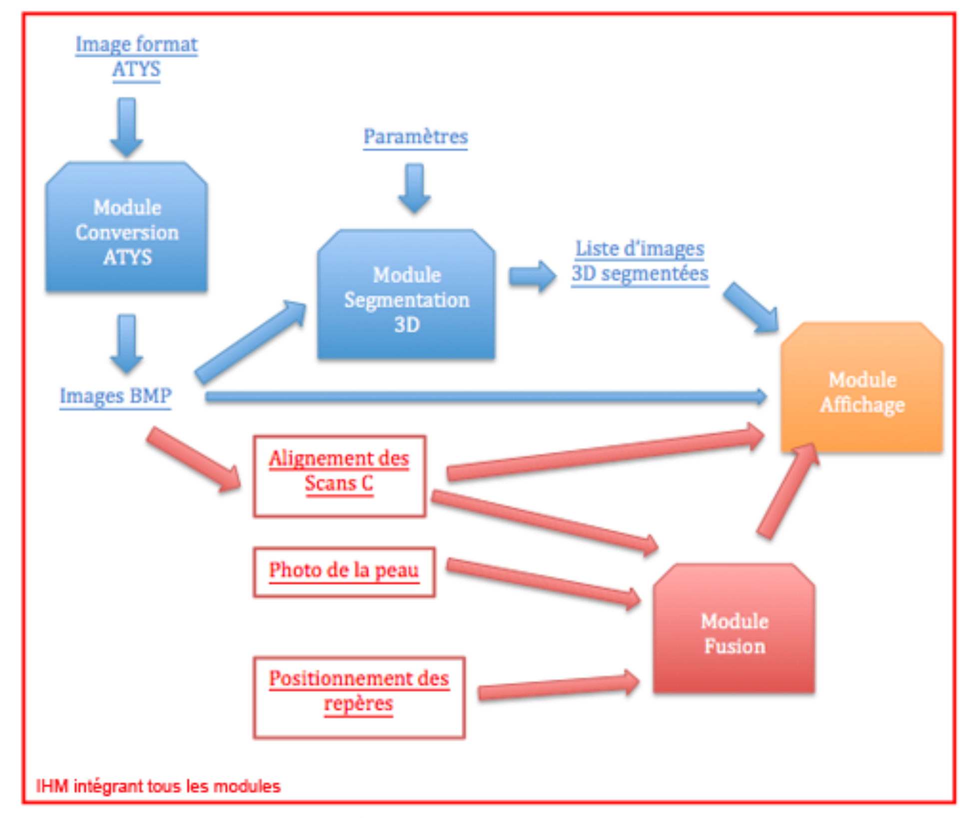


Figure 2 - Intégration des modules dans l'existant

PARTIE 1 : RESSORTIR LE MODULE D'AFFICHAGE 3D

1 - Présentation de l'application existante

L'application existante est composée de nombreux modules, elle est donc très riche et un fichier comme ImageTroisD.cpp comporte plus de 15000 lignes de code.

Néanmoins, seule une partie de cette application m'intéresse, et il s'agit de la partie « affichage 3D », qui n'est autre que la représentation 3D de l'acquisition. Sur la photo ci-dessous, il faudra donc isoler la partie gauche contenant l'acquisition 3D ainsi que le menu de droite permettant à l'utilisateur de se balader dans l'acquisition.

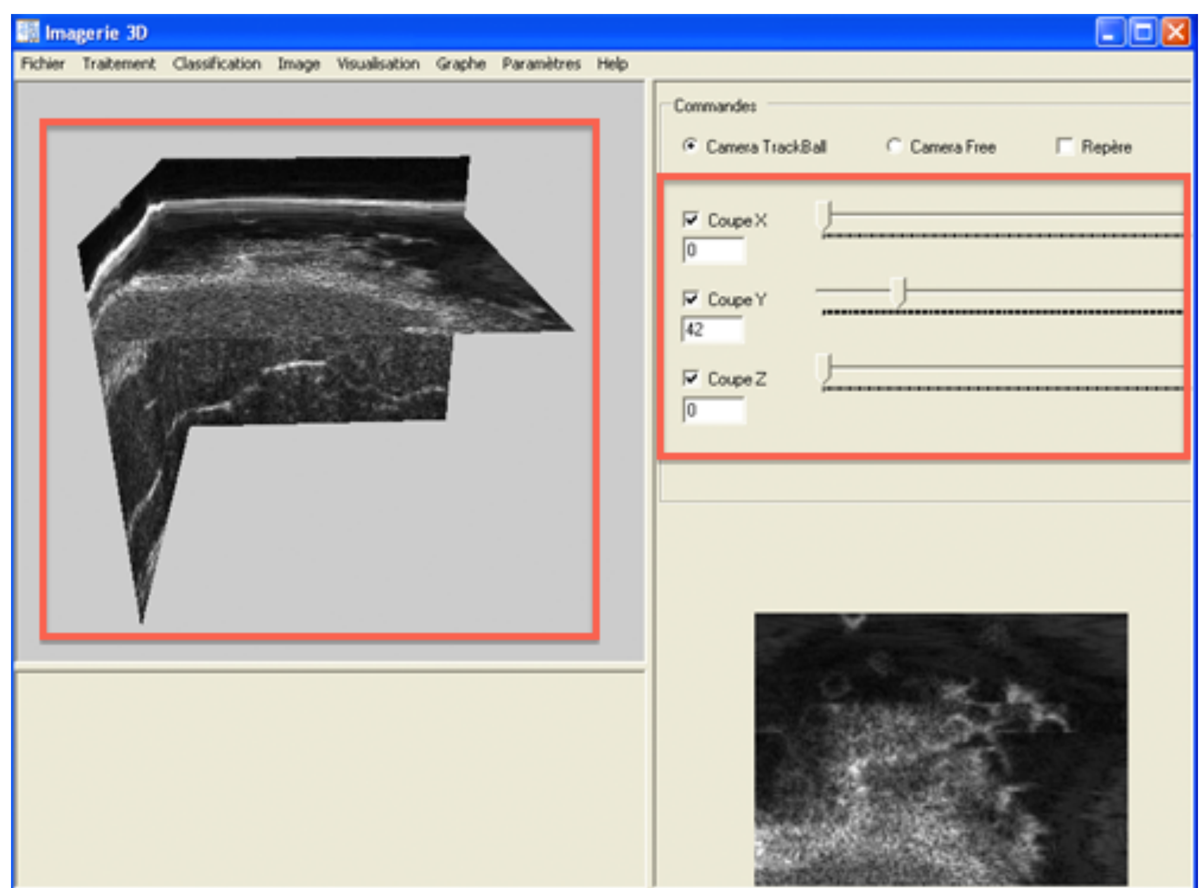


Figure 3 - Module d'affichage de l'application existante (en rouge)

2 - Premières difficultés rencontrées

La première difficulté du projet est survenue lors de l'étude du code existant. En effet, bien que l'application existante soit très complète au niveau des fonctionnalités implémentées, or le projet ne contenait aucune modélisation, il m'était très difficile de ne ressortir que la partie qui m'intéressait, soit la partie « affichage de l'acquisition 3D ». Il aura fallu plusieurs tentatives, que je vais décrire juste après, avant de finalement s'orienter vers une nouvelle application sans se préoccuper de l'existant.

Petit rappel avant de décrire les différentes tentatives de compréhension du code existant, il fallait également faire fonctionner ce module d'affichage 3D sous Visual Studio 2010, version plus récente.

Tout d'abord, la première étape pour réaliser un projet sous Visual Studio 2010 est de pouvoir configurer celui-ci de telle sorte que tous les liens vers les librairies nécessaires soient bons et ne posent donc pas de soucis lors de la compilation. Pour cela, j'ai réalisé une aide à la configuration d'un projet utilisant OpenGL (voir Annexe 1).

La première tentative, avant de pouvoir isoler le module d'affichage, a été de passer le projet sous Visual Studio 2010. L'outil automatique de Visual Studio 2010 pour transformer un projet issu de 2005 en un projet 2010 retourne de nombreuses erreurs à la compilation et ne fonctionne donc pas. Après avoir passé un jour de débogages, sans succès, j'ai décidé de changer de technique d'approche pour ne pas perdre trop de temps.

La seconde tentative a été de ressortir une modélisation de l'application existante. Pour cela j'ai donc épluché toutes les classes pour trouver les liaisons (grâce aux créations d'objets), mais le résultat obtenu m'a directement emmené à la conclusion que je ne pouvais rien retirer de l'application existante.

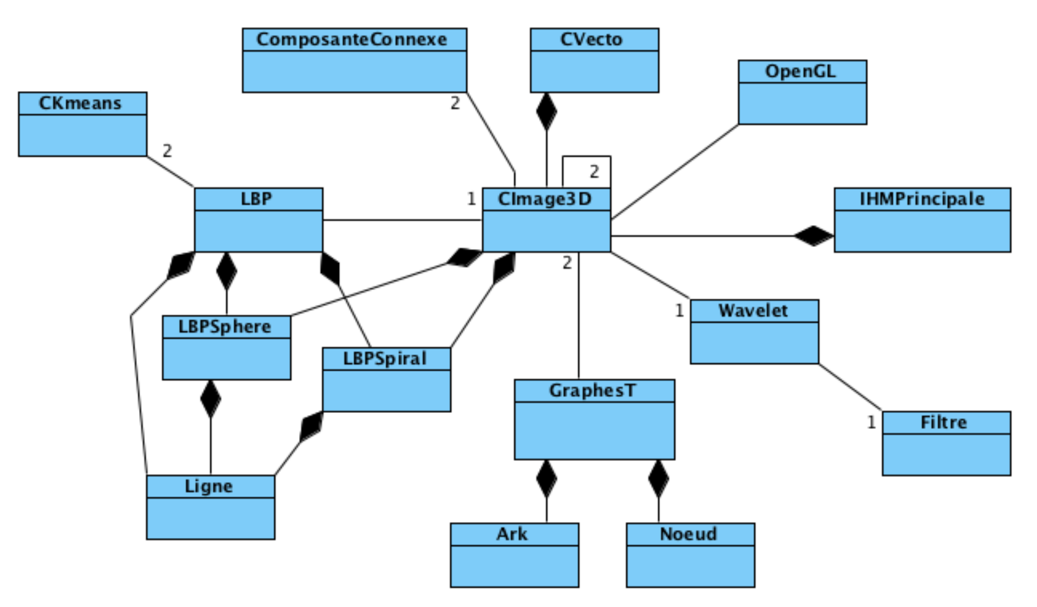


Figure 4 - Diagramme de classe (Application existante)

3- Nouvel objectif

Suite à une réunion avec les encadrants et les clients, nous avons donc décidé de totalement changer d'objectif : il faut impérativement créer une application basée sur une modélisation claire et qui devra rendre l'application modulable et indépendante d'OpenGL. Ainsi quiconque reprendra le projet devra comprendre mon raisonnement et la construction de l'application à travers cette modélisation.

Mr. Baudouin MARTIN a été une aide précieuse tout au long de la modélisation du projet puisqu'il m'a apporté beaucoup de conseils sur la réalisation des diagrammes, leurs utilités et ce que je devais en déduire pour mon application.



Mais avant de construire tous ces diagrammes, il a d'abord fallu que je comprenne le fonctionnement d'OpenGL lors de l'affichage d'une acquisition 3D.

4- Apprendre le fonctionnement d'OpenGL

Pour pouvoir réaliser une modélisation correcte de l'application, il était impératif de comprendre le fonctionnement d'OpenGL pour en ressortir tous les objets nécessaires ainsi que toutes les fonctions. Tout cela dans le but de pouvoir réaliser une modélisation correspondant parfaitement au fonctionnement d'OpenGL pour mon module d'affichage 3D.

Après avoir terminer plusieurs tutoriels, j'ai finalement commencé à refaire la partie « affichage d'une acquisition 3D » sous Visual Studio 2010, en repartant de zéro et donc sans se préoccuper de l'application existante. Et c'est juste avant la reprise de janvier que j'ai finalement réussi à récréer ce module.

PARTIE 2 : MODELISATION DE L 'APPLICATION

Pour cette partie, de nombreuses réunions avec le client et les encadrants ont eu lieu afin de permettre la bonne évolution des diagrammes que je réalisais. En effet, ces diagrammes traduisant notre vision de l'application et plus précisément comment nous souhaitons la concevoir, il était impératif de se voir régulièrement pour discuter autour de ces diagrammes.

La modélisation du projet aura duré un peu plus d'un mois, mais la structure de l'application est déjà bien avancée grâce à cette modélisation.

1- La démarche

En accord avec les encadrants et avec le client, plusieurs objectifs sont ressortis de cette partie modélisation.

Le premier étant d'avoir une application modulable puisque celle-ci va être amenée à évoluer dans un futur proche, il faut donc la rendre la plus compréhensible possible pour que quiconque comprenne le raisonnement lors de la conception de l'application. Cela passe donc obligatoirement par la réalisation de différents diagrammes tels que le diagramme de cas d'utilisation, les diagrammes de séquences et le diagramme de classe.

Le second objectif était d'isoler OpenGL de l'application, pour que celle-ci puisse fonctionner sans avoir besoin de l'affichage 3D, puis pour que l'outil d'affichage 3D qui est aujourd'hui OpenGL, puisse être modifié ou remplacé facilement. En d'autres mots, il faut rendre l'application indépendante d'OpenGL.

Pour palier à ces deux objectifs, il faudra alors mettre en place le patron Modèle-Vue-Contrôleur (MVC) qui permettra d'avoir une application modulable, évolutive, indépendante d'OpenGL et dont la partie affichage sera séparée des données.

Je rappelle que le modèle MVC permet de séparer les données de l'affichage, sachant que le modèle contient les données, la vue représente l'affichage tandis que le contrôleur permet de faire la liaison entre les deux. Voici le diagramme de classe d'un MVC :

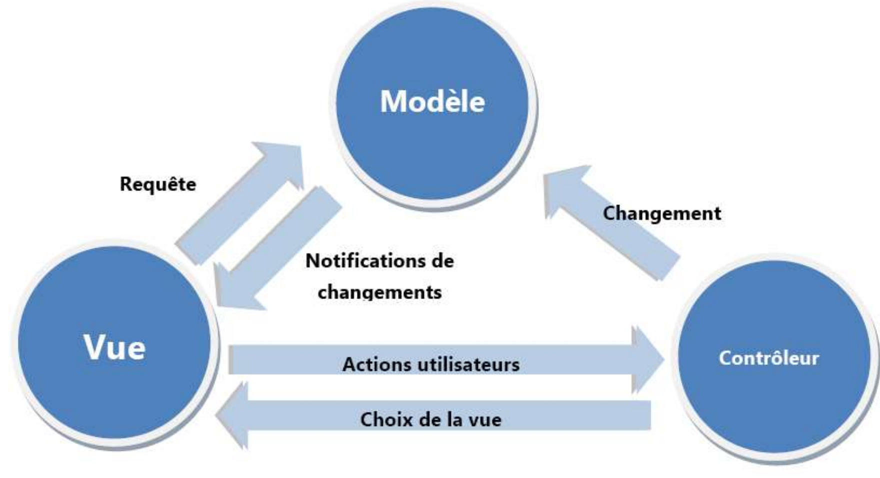


Figure 5 - Modèle MVC



2 – Le diagramme de cas d'utilisations

La première étape de la modélisation est de ressortir les résultats que le client attend de la future application, en une question simple : « quels résultats cette application permet-elle d'obtenir ? ».

Je vais vous présenter l'évolution du diagramme de cas d'utilisation en détaillant mes premières erreurs pour enfin parvenir à la version finale.



Figure 6 - Diagramme de cas d'utilisation (v1)

Il s'agit de la première version du diagramme de cas d'utilisation bien avant que l'objectif de modélisation n'arrive puisqu'il s'agit du diagramme présent dans la première version du cahier de spécifications.

L'erreur majeure de ce diagramme est d'avoir confondu les actions possibles de l'application avec les résultats que celle-ci doit produire. De plus, l'architecture du diagramme n'est pas correcte puisque les cas d'utilisation doivent appartenir à un système. Avec la version 2, cette première erreur est en

partie effacée même si ce concept de résultat avait encore du mal à rentrer. Ainsi, nous y retrouvons encore des fonctionnalités comme « positionner le gabarit3D », qui sont des actions de l'utilisateur et non un résultat attendu. De plus, l'action « gérer les gabarits » n'est également pas un résultat attendu puisqu'il ne fournira pas un affichage ou autre d'une modification d'un gabarit par exemple.

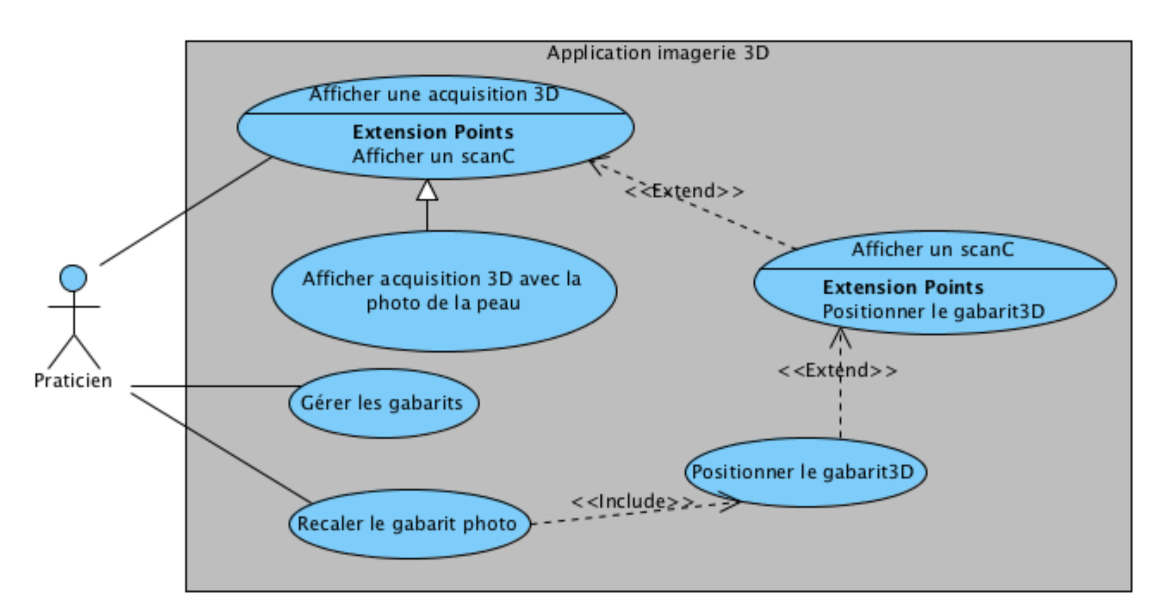


Figure 7 - Diagramme de cas d'utilisation (v2)

Et c'est la troisième version qui va permettre de répondre aux exigences d'un diagramme de cas d'utilisation, puisque celle-ci ne fait ressortir que les résultats que l'application permettra d'obtenir. Ces résultats sont donc l'affichage d'une acquisition, le recalage et le redimensionnement de la photo de la peau ainsi que l'affichage de l'acquisition fusionnée avec la photo de la peau.

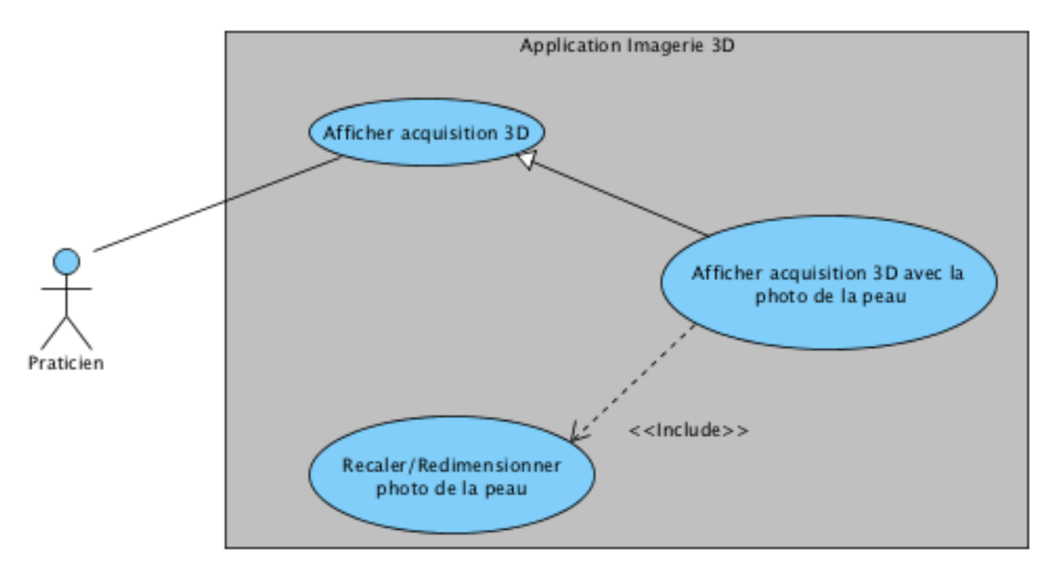


Figure 8 - Diagramme de cas d'utilisation (v3)



3 – Les diagrammes de séquences

Cette étape de la modélisation a été la plus importante puisqu'elle m'a permis de ressortir tous mes objets, toutes les fonctions et tous les échanges entre les objets de mon application. Grâce à ces diagrammes, j'ai pu déduire une grosse partie du diagramme de classe et j'ai également pu définir les interfaces homme-machine (IHM) avec le client puisque tous ces diagrammes représentent le déroulement de l'application lorsque l'utilisateur déclenche une action.

Ces diagrammes représentent la manière dont l'application va être fabriquée. Ils reprennent tous les enchaînements et échanges entre les différents objets, tout cela à travers des appels de fonctions et des instanciations d'objets. Il s'agit tout simplement de la vie de l'application et en fait donc une partie très importante de la modélisation et qui m'a également pris beaucoup de temps.

Ainsi, j'ai réalisé sept diagrammes de séquences, chacun représentant une action bien précise. Je vais d'abord commencer par présenter le diagramme qui a connu le plus de changement au cours de la modélisation, puis j'enchaînerai sur les autres sans reprendre leurs évolutions puisqu'elles sont minimes ou parce qu'elles correspondent à celles que je vais présenter pour le premier diagramme de séquence.

- Diagramme 1 : « Afficher une acquisition 3D »

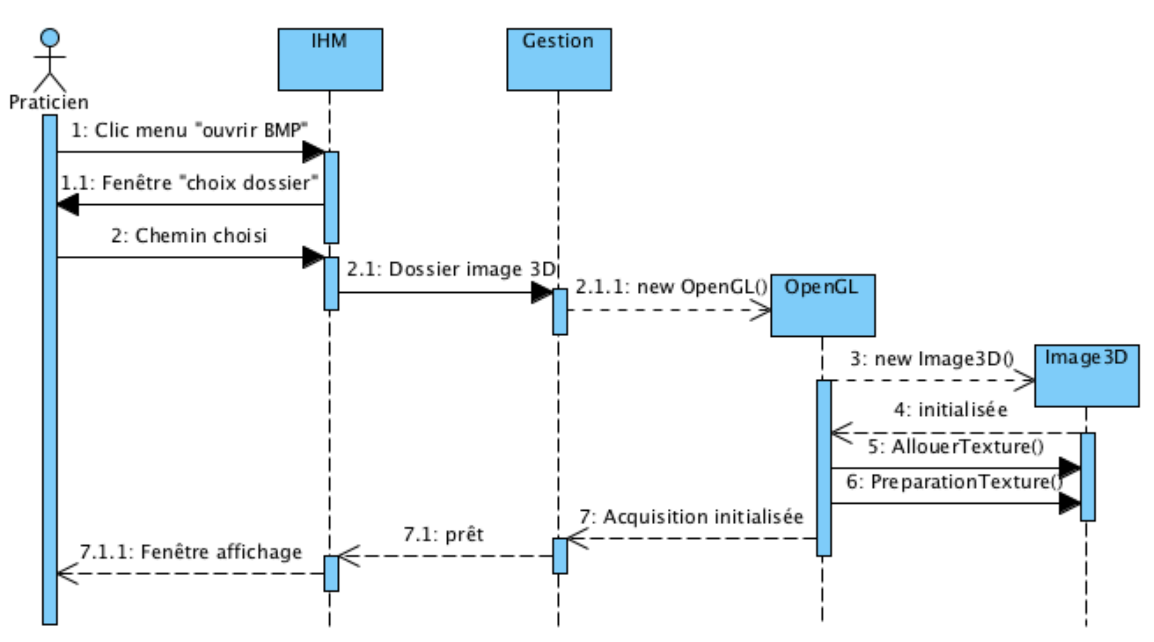


Figure 9 - Diagramme de séquence (afficher acquisition 3D, v1)

Dans cette première version, le principal problème est le positionnement de l'objet OpenGL ainsi que son interaction avec l'objet « Image3D ». En effet, si nous regardons le diagramme, nous en déduisons que l'acquisition 3D est contenue dans OpenGL alors que cela ne devrait surtout pas être le cas puisque nous souhaitons isoler OpenGL pour rendre l'application et nos données indépendante de cet outil d'affichage.

La correction apportée a donc été de repositionner OpenGL et de créer l'acquisition 3D depuis le module de Gestion et non pas depuis OpenGL. De plus, ce n'est pas Gestion qui doit créer la fenêtre

d'affichage 3D mais plutôt l'IHM principale puisque celle ci contient la fenêtre d'affichage 3D. Voici donc la dernière version de ce diagramme de séquence :

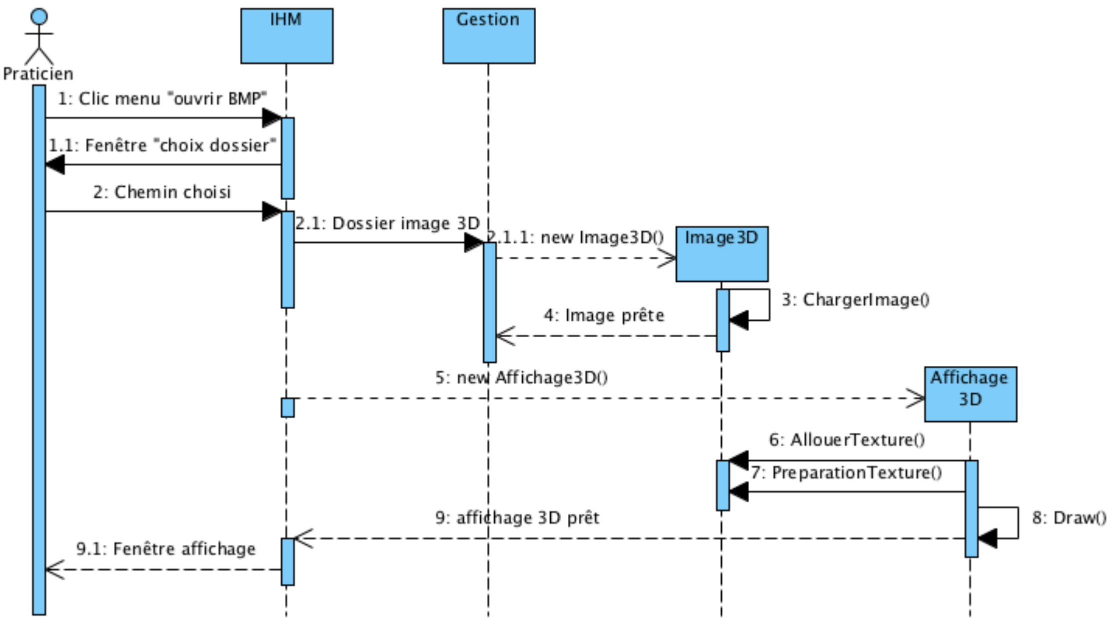


Figure 10 - Diagramme de séquence (Afficher une acquisition 3D, v2)

Ainsi, l'acquisition 3D est d'abord créée par le module de Gestion, elle est ensuite chargée en mémoire grâce à la fonction « ChargerImage() ». Une fois l'image prête, le module de Gestion va créer la fenêtre d'affichage 3D qui recevra en paramètre l'acquisition à afficher. Pour charger les textures, l'objet « affichage3D » ira directement chercher dans « Image3D » les informations dont il a besoin pour dessiner l'acquisition.

Concernant l'IHM de notre application, ce diagramme nous apprend simplement que nous aurons besoin d'un menu « ouvrir » pour que l'utilisateur sélectionne le dossier contenant les images BMP de l'acquisition 3D.

- Diagramme 2 : « Recalage de l'acquisition 3D»

Le recalage de l'acquisition 3D correspond à l'algorithme qui va permettre de recaler l'acquisition de telle sorte que la surface de la peau soit parallèle aux scans C. Cette étape intervient juste après que l'image ait été chargée et juste avant que le module de Gestion ne crée la fenêtre d'affichage 3D (cf. le diagramme de séquence précédent).

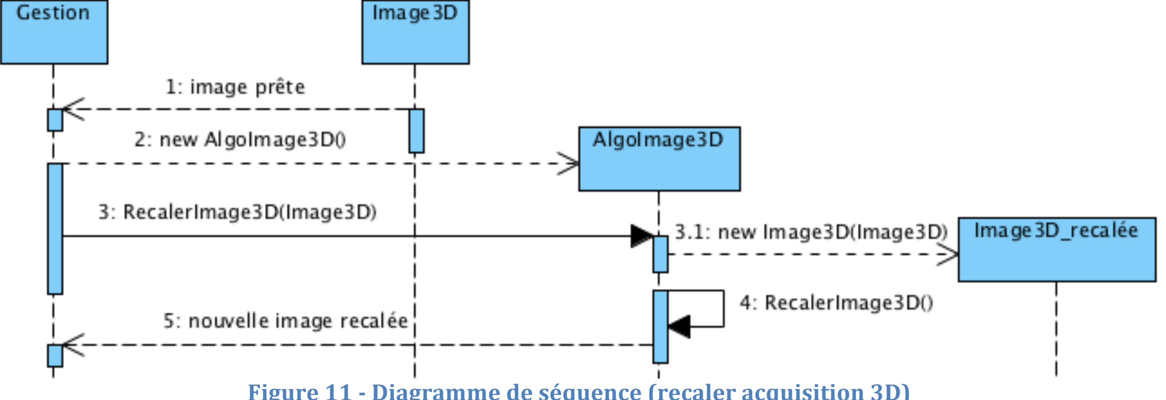


Figure 11 - Diagramme de séquence (recaler acquisition 3D)



Ce diagramme sera amener à évoluer, notamment au niveau de ce que la fonction « recalerImage » retournera une fois qu'elle sera implémentée. Pour l'instant, cette fonction retourne une nouvelle image recalée.

- Diagramme 3 : « Mise à jour de l'affichage de l'acquisition 3D et afficher un scanC »

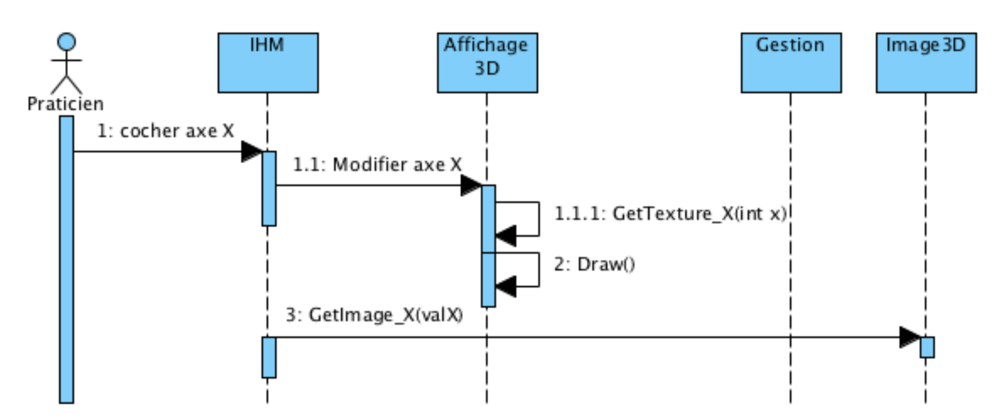


Figure 12 - Diagramme de séquence (MAJ acquisition 3D et afficher scan C)

La mise à jour de l'affichage de l'acquisition 3D est démarrée par une action de l'utilisateur qui va modifier la valeur d'un des axes de l'acquisition 3D. Cette action va entraîner la recherche des textures correspondant à la nouvelle valeur de l'axe X (par exemple) dans la classe « Affichage 3D » grâce à la fonction GetTexture_X(int valx). Il n'y a pas d'échange avec la classe « Image3D » puisque les textures ont déjà été chargées lors de la création de la fenêtre d'affichage 3D.

De plus, l'action de l'utilisateur va également lancer l'appel de la fonction GetImage_X(int valx) qui va aller chercher dans la classe « Image3D » les valeurs des pixels permettant d'obtenir le scan C correspondant.

Concernant l'IHM de l'application, ce diagramme nous confirme uniquement l'intérêt des scroll permettant de modifier les valeurs des axes de l'acquisition 3D.

- Diagramme 4 : « Modification d'un gabarit »

Je rappelle qu'un gabarit contient les coordonnées des points repères ultrasonores et des points repères de la photo de la peau. Les dimensions des translations sont donc connues et seront directement recalculées lors de la modification d'une des coordonnées d'un gabarit.

Je n'ai pas ajouté les diagrammes de séquences des actions « Ajouter un gabarit » et « Supprimer un gabarit » puisque ces diagrammes sont presque similaires à celui présenté ci-dessous.

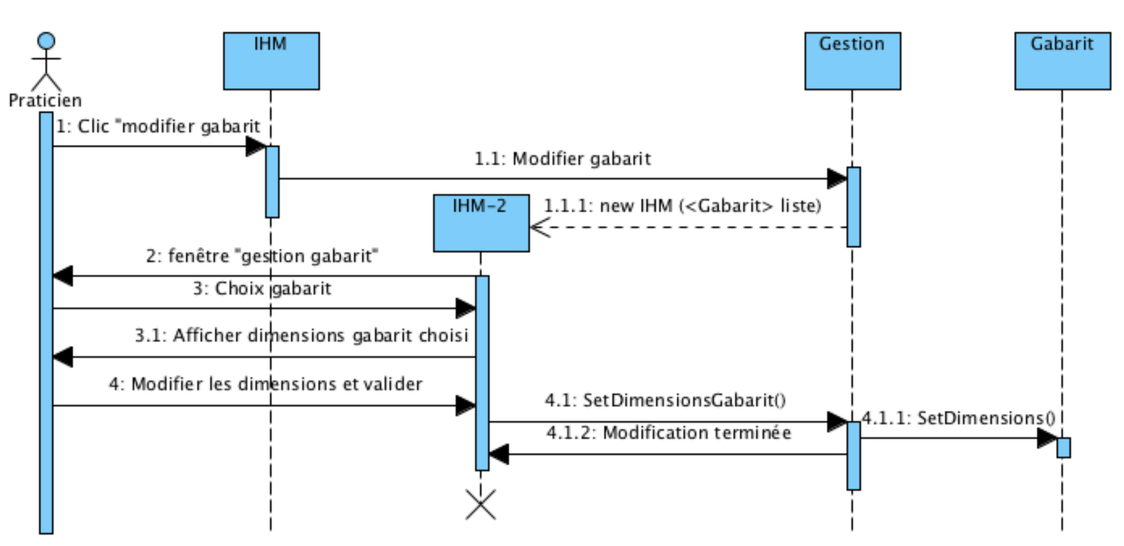


Figure 13 - Diagramme de séquence (modifier un gabarit)

Sur ce diagramme, le rôle de la classe « Gestion » est particulièrement mis en avant puisque lorsque l'utilisateur a saisi les nouvelles coordonnées du gabarit et valider celles-ci, ces nouvelles données sont transmises à la classe Gestion qui va alors demander à la classe Gabarit de mettre à jour ses coordonnées. L'IHM n'interagit donc pas du tout avec les données car tout passe par Gestion lorsqu'une donnée est modifiée.

Concernant l'IHM principale, ce diagramme de séquence me permet de savoir que je vais devoir définir une première fenêtre pour que l'utilisateur choisisse le gabarit à modifier, puis une seconde fenêtre à partir de laquelle l'utilisateur pourra modifier les coordonnées. Ce diagramme nous permet également de savoir qu'un nouvel onglet « Gestion gabarit » sera dans la barre de menu.

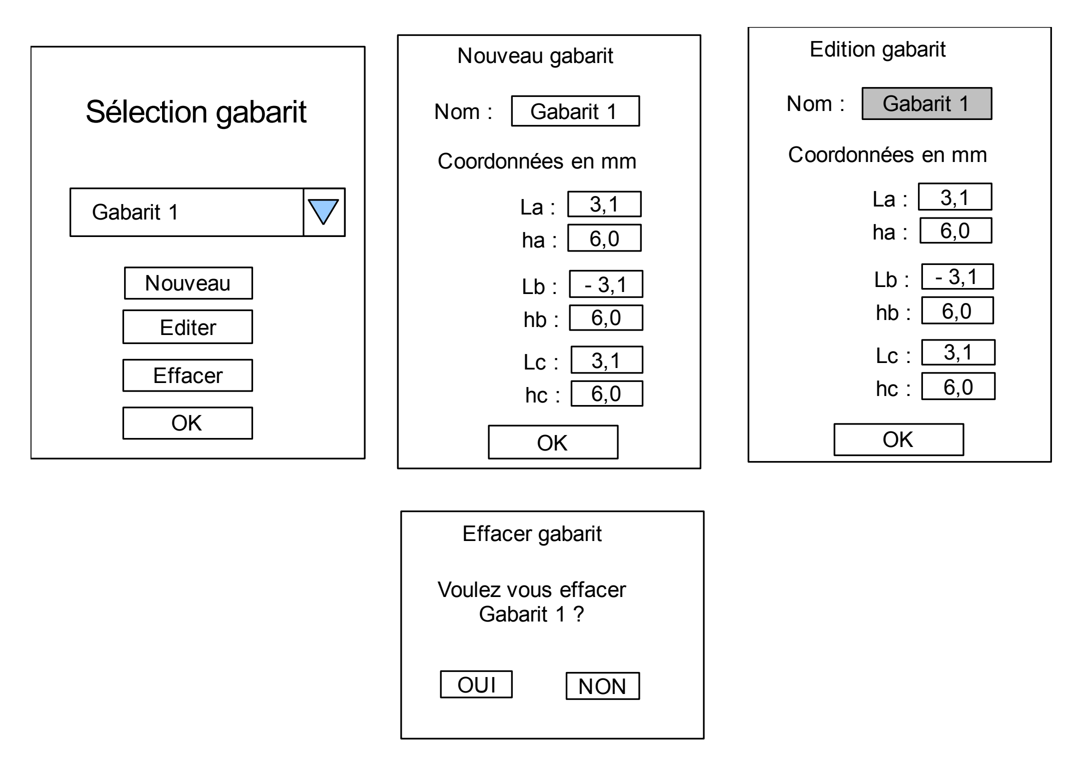


Figure 14 - Proposition d'interface "Gestion des gabarits"



- *Diagramme 5* : « Positionner les points repères sur la photo de la peau »

Ce positionnement de point repère sera différent de celui des points repère de l'acquisition 3D car pour la photo je n'ai besoin que des coordonnées X et Y. Les coordonnées Z de la photo ne seront connues que lors de la fusion avec l'acquisition 3D, pour l'instant ces coordonnées sont nulles.

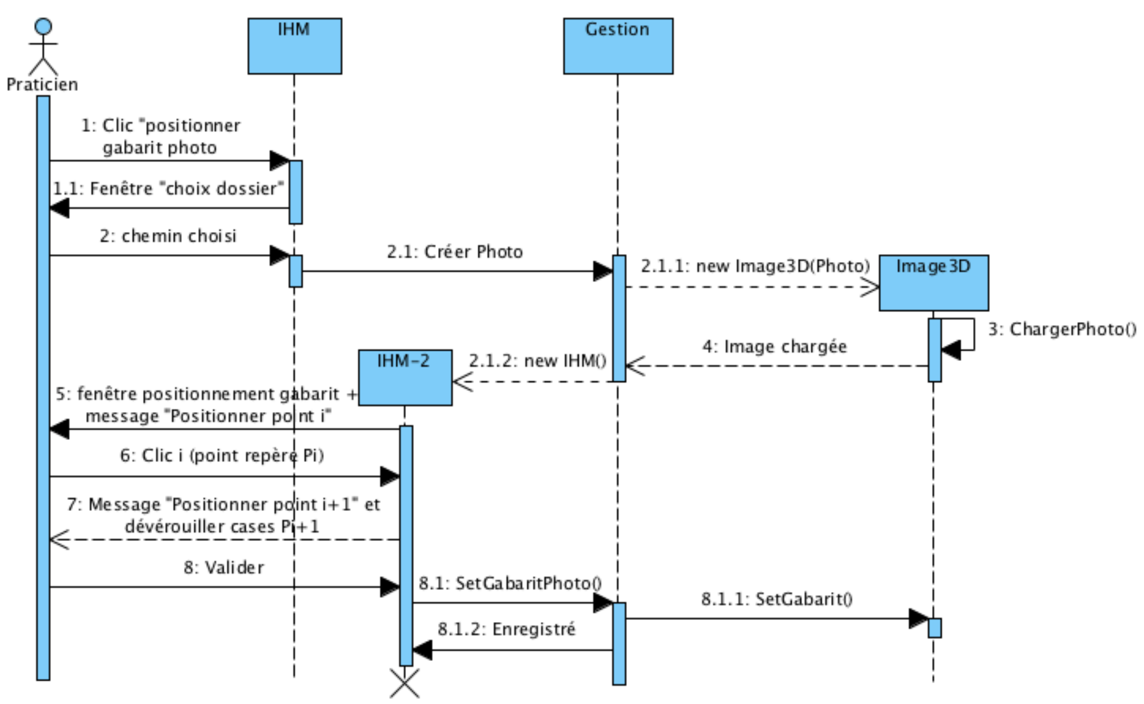


Figure 15 - Diagramme de séquence (Positionner gabarit photo)

Encore une fois, nous pouvons observer le rôle de Gestion qui va faire le lien entre l'IHM et les données du gabarit à mettre à jour. La séparation entre IHM et données est donc parfaitement respectée.

Nous remarquons également la création d'une nouvelle image 3D qui correspond à la photo de la peau et dont la profondeur sera initialisée à 0. L'action de l'utilisateur qui déclenche ce diagramme est le clic sur le menu « Positionner gabarit photo ». Un point important qui avait été souligné par le client, était de savoir quel point l'utilisateur devait saisir en premier, en second etc., pour cela, j'ai décidé de n'activer qu'un seul des champs représentant les coordonnées des points. Par exemple, au début de la saisie, l'application demandera à l'utilisateur de saisir le point 1, et seules les coordonnées de ce point pourront être modifiées jusqu'à ce que l'utilisateur valide sa saisie.

Concernant l'IHM, ce diagramme nous permet de ressortir l'interface de la fenêtre de positionnement d'un gabarit de la photo de la peau. En effet, nous devons avoir la photo affichée, ainsi que des champs pour afficher les coordonnées saisies par clic de la souris par l'utilisateur. De plus, l'interface devra interagir avec le praticien pour le guider sur l'ordre des points de repère qu'il devra saisir. Une légende des points repère devra également apparaître sur la fenêtre pour guider un peu plus l'utilisateur lors du positionnement des points.

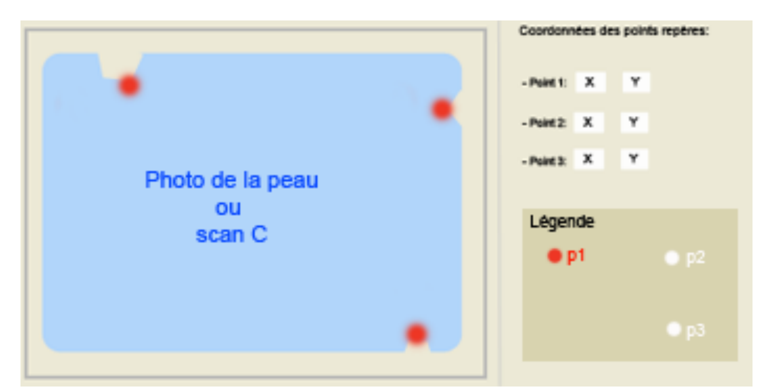


Figure 16 - Aperçu de l'interface "positionner points repères"

- Diagramme 6 : « Positionner le gabarit de l'acquisition 3D »

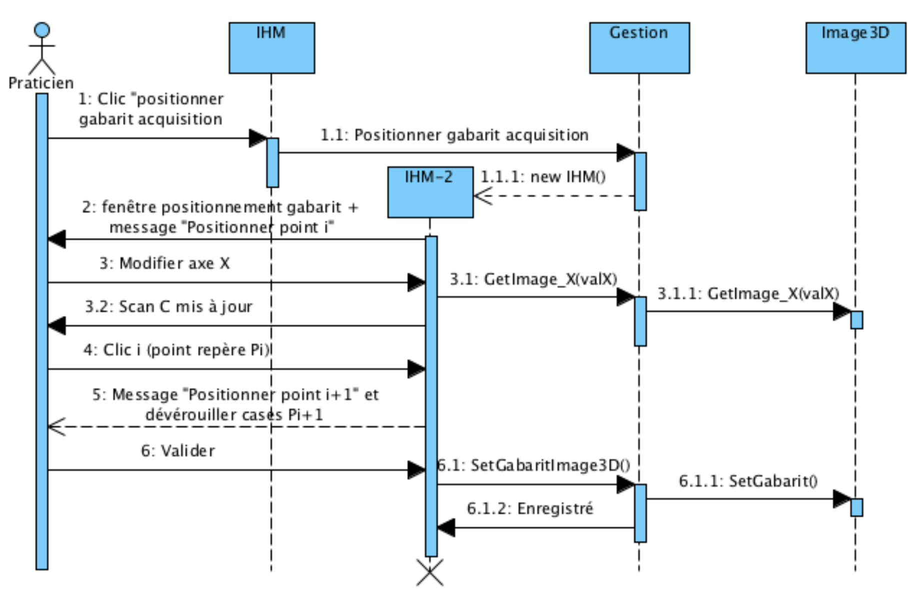


Figure 17 - Diagramme de séquence (Positionner gabarit acquisition)

La principale différence de ce diagramme avec le précédent est que l'utilisateur va devoir positionner des points repères sur l'acquisition 3D à l'aide des scans C. Pour cela, l'utilisateur souhaite pouvoir modifier n'importe quand le scan C sur lequel il va rechercher et positionner un point repère. Il doit y avoir une interaction avec l'affichage d'un scan C et cette fois-ci, la profondeur aura donc un rôle dans le positionnement des points repères.

L'option supplémentaire qui permet de modifier le scan C sur lequel l'utilisateur doit placer les points repères est donc ajoutée et déclenchée lorsque l'utilisateur modifie la valeur de l'axe Z de l'acquisition 3D. Le déroulement engendré est le même que pour la mise à jour du scan C sur l'IHM principale avec l'appel de la fonction GetImage_Z(ind z) qui va mettre à jour les pixels du scan C en fonction des valeurs des axes.

Concernant l'IHM de l'application, ce diagramme me permet de ressortir la même chose que pour le positionnement des points repères de la photo, avec l'ajout du scroll Z permettant de modifier l'image (le scan C) affichée sur la fenêtre.

- *Diagramme 7* : « Fusionner la photo de la peau et l'échographie 3D »

L'affichage de la fusion de la photo de la peau avec l'échographie 3D consiste uniquement à afficher la photo de la peau avec les points repères correspondant, ainsi que l'affichage de la fusion. La fusion n'est autre qu'une superposition de la photo de la peau avec un scan C de l'échographie 3D, dont la transparence peut varier selon les envies de l'utilisateur.

Comme cela sera expliqué dans la suite de ce rapport, l'unique manipulation réalisée dans ce module est la découpe de la photo de la peau, ce qui formera une nouvelle photo, de la taille de l'échographie et qui contiendra tous les points repères.

L'algorithme correspondant à cette opération est situé dans la classe AlgosImage3D et doit donc être appelé pour que celui-ci puisse retourner une nouvelle photo.

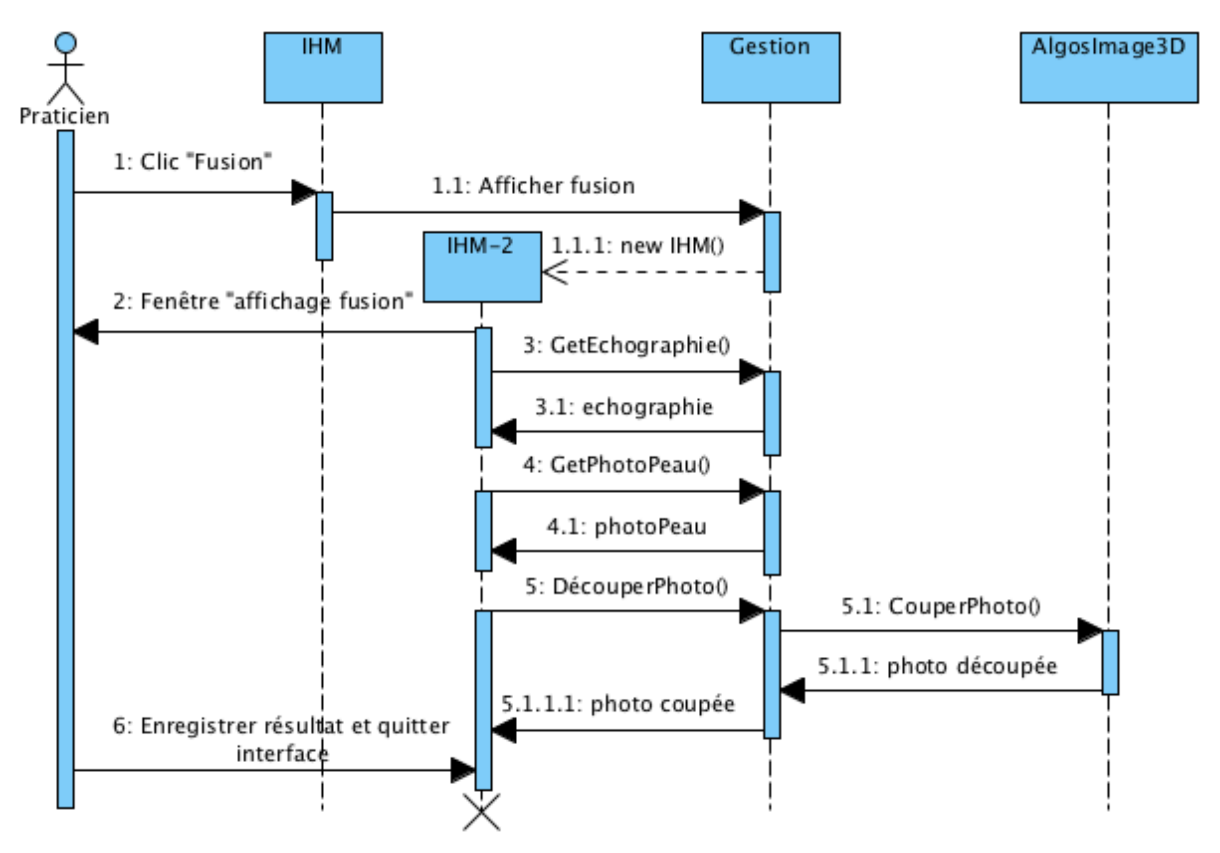


Figure 18 - Diagramme de séquence: fusion

4 – Le diagramme de classe

Nous voici enfin à la dernière étape de la modélisation, celle qui va permettre de poser une structure dans notre application. Structure qui doit impérativement être modulable, répondre aux obligations d'un MVC et isoler OpenGL.

Il y a eu beaucoup de discussions autour de ce diagramme de classe qui a par conséquent connu plusieurs versions avant de voir la version « finale » me permettant de passer à l'étape développement du projet. A noter que cette version du diagramme de classe va très certainement connaître quelques modifications infimes dans la suite du projet puisque des fonctions viendront s'ajouter à celui-ci.

Je vais maintenant présenter les différentes versions du diagramme de classe et expliquant les raisons de ses évolutions.

Version 1 :

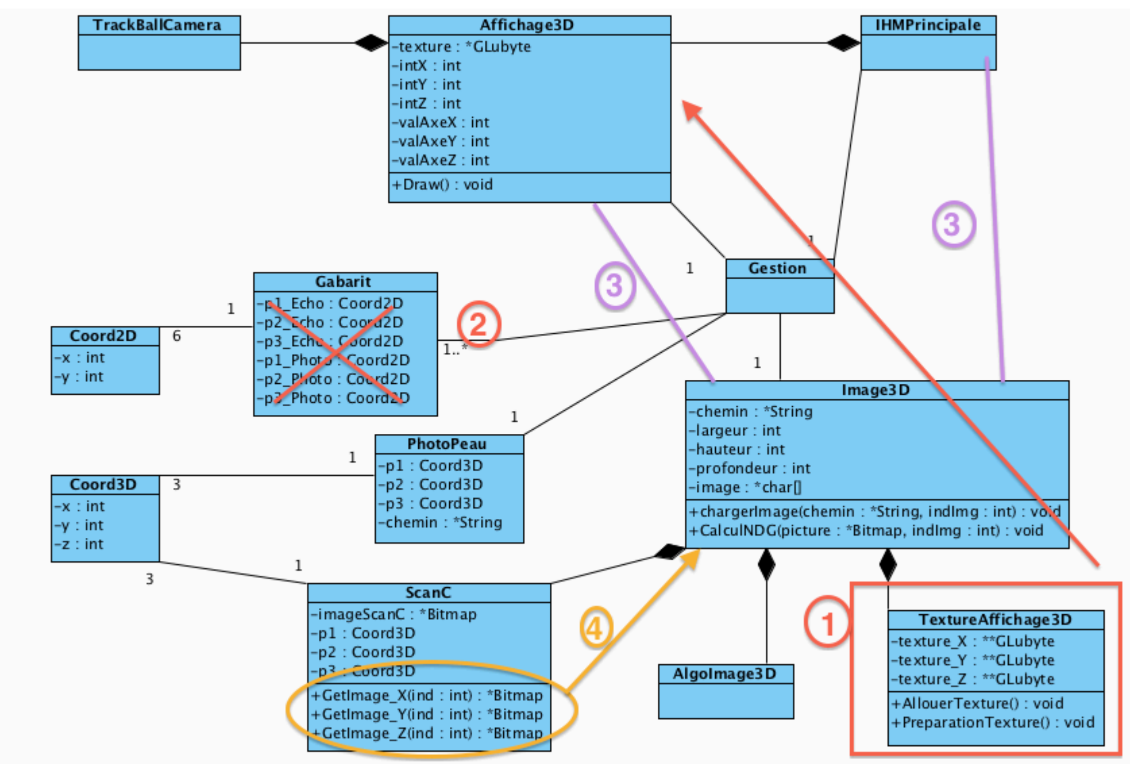


Figure 19 - Diagramme de classe (v1)

La partie supérieure de ce diagramme (TrackBallCamera, Affichage3D et IHMPrincipale) représente la partie « Vue » du modèle MVC, la classe « Gestion » correspond au « Contrôleur » du MVC tandis que toutes les autres classes représentent le « Modèle » du MVC.

Nous pouvons d'abord constater la présence d'OpenGL dans la partie « Modèle » et notamment dans la classe « TextureAffichage3D » alors que l'objectif principal de la modélisation à mettre en place est de rendre indépendant l'application d'OpenGL. Dans cette configuration, ce n'est pas le cas et il faut donc impérativement placer ces textures dans la classe « Affichage3D » de la Vue. C'est donc



« Affichage3D » qui doit venir demander à « Image3D » les informations dont il a besoin sur l'image pour obtenir ces textures OpenGL.

La seconde erreur se trouve au niveau des attributs des classes « Gabarit » « PhotoPeau » et « ScanC », puisque les attributs correspondant aux coordonnées ne doivent pas apparaître dans le diagramme de classe puisque la liaison suffit. Erreur d'étourderie corrigée.

L'erreur numéro 3, qui représente un oubli de liaison entre l'IHM et les données. En effet, lorsque l'IHM souhaite afficher l'acquisition 3D, elle va directement chercher les informations nécessaires dans la classe « Image3D ».

Enfin, la dernière erreur, concerne toute la classe « ScanC » puisque les fonctions qui servent à mettre à jour le scanC (Bitmap) vont simplement rechercher les pixels nécessaires dans le tableau contenant tous les pixels de l'image. Cette classe doit donc faire partie de « Image3D ».

Version 2 :

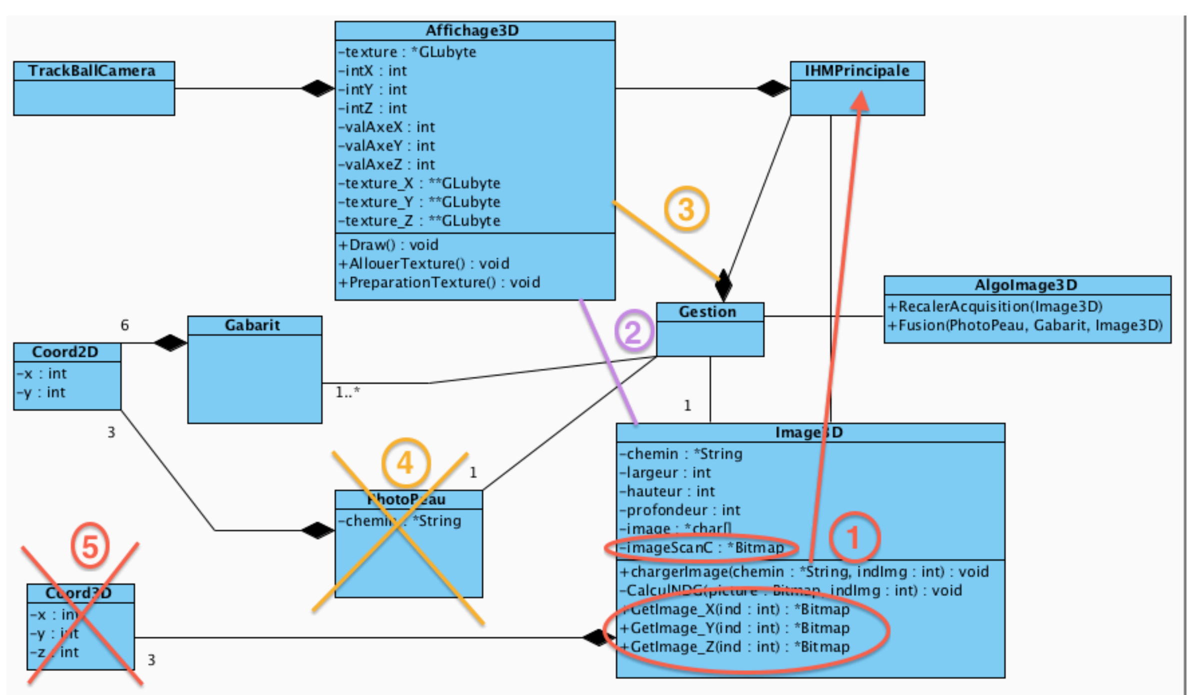


Figure 20 - Diagramme de classe (v2)

L'objet Bitmap présent dans la classe « Image3D » représente le scanC qui sera affiché en bas à droite de notre application. Il est construit en fonction de la position des axes X, Y et Z, ainsi lorsque la valeur d'un des axes est modifiée. Par conséquent cette image ne doit pas se trouver dans la partie Modèle du MVC mais dans la partie Vue puisqu'à chaque modification d'un axe, l'IHM va aller chercher les pixels correspondant dans le tableau de pixels de « Image3D ».

La seconde erreur concerne le lien entre « Affichage3D » et « Image3D, il s'agit d'un oubli puisque cette remarque avait déjà été faite lors de la première version du diagramme de classe.

La troisième remarque est également due à un lien manquant entre « Affichage3D » et « Gestion ». Ce lien de composition est obligatoire puisque notre partie Contrôleur du MVC doit pouvoir accéder directement à « Affichage3D » si notre acquisition 3D est modifiée.

Quatrième erreur, la classe « PhotoPeau », qui représente la photo de la peau prise par le praticien. Cette photo sera finalement traitée comme une acquisition 3D et sera donc assimilée à la classe « Image3D ». En plus des points x et y de la photo, la profondeur de la photo de la peau sera d'abord alors initialisée à 0.
Dernière remarque sur cette seconde version du diagramme de classe, la classe « Coord3D » qui représente les points repères qui seront positionnés sur l'acquisition 3D. Ces points correspondent à un gabarit, la classe « Image3D » sera donc reliée à « Gabarit ».

Nous pouvons également remarquer l'ajout de nouvelles fonctions dans la classe « AlgoImage3D », qui correspondent aux deux modules à développer pour l'application (le recalage de l'acquisition 3D ainsi que la fusion).

Version « finale » :

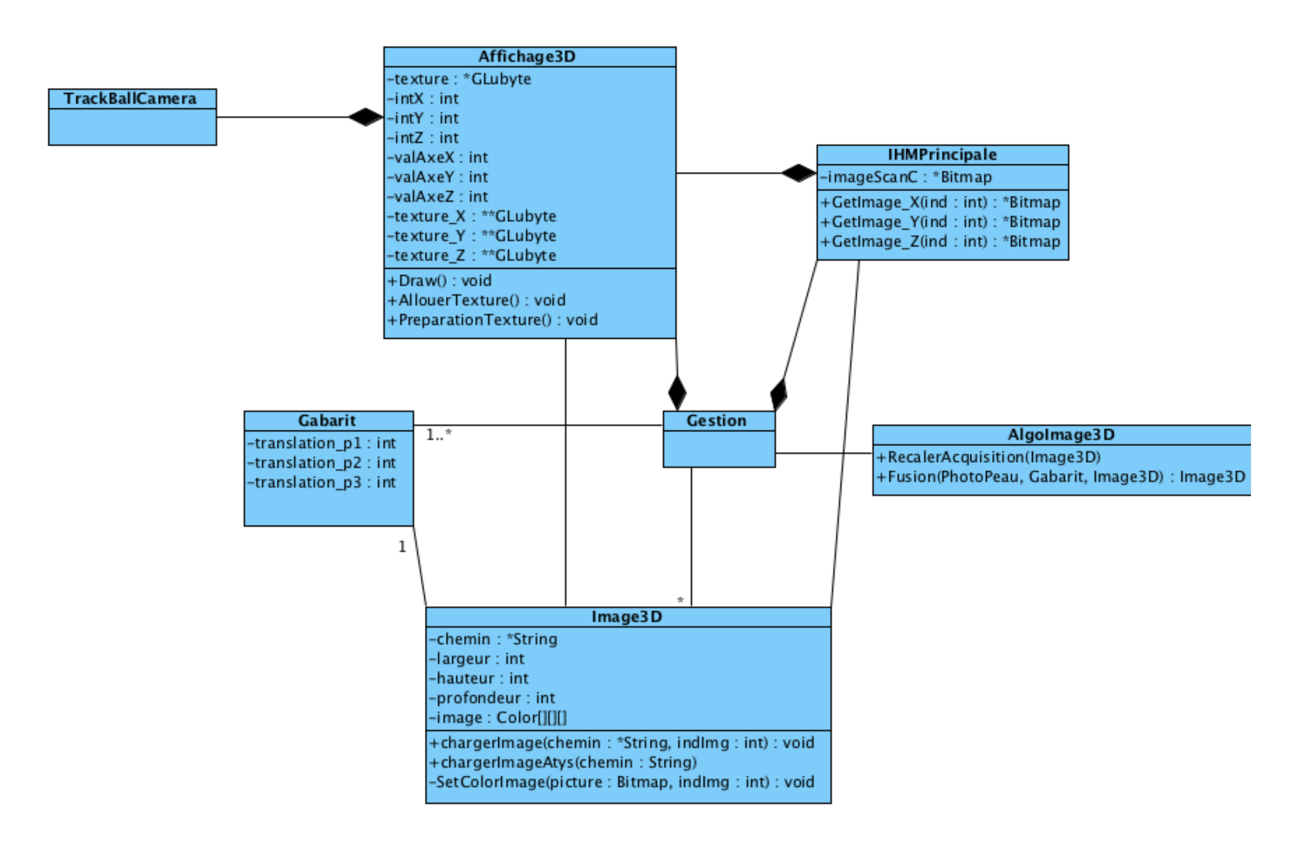


Figure 21 - Diagramme de classe (v3)

Voici donc la troisième version du diagramme de classe. Il sera néanmoins amené à évoluer avec l'ajout de nouvelles fonctions dans certaines classes, notamment la classe « AlgoImage3D ».

CONCLUSION MODELISATION

Cette partie modélisation m'a permis de mieux cerner le sujet ainsi que les attentes du client et j'ai donc pu réalisé une structure cohérente de la future application. Cette étape est très importante dans un projet puisqu'elle en est la base et permet de rendre l'application très modulable et facile à comprendre et à reprendre pour n'importe qui.

PARTIE 3 : DEVELOPPEMENT DES INTERFACES

Pour le développement des interfaces, j'ai commencé par réaliser une première version en fonction de ce que nous avions défini dans le cahier de spécification système avec le client. Je savais qu'il faudrait retravailler sur ces versions et c'est pourquoi j'ai rencontré le client pendant à plusieurs reprises pour pouvoir apporter les modifications nécessaires.

Encore une fois le dialogue avec le client est très important, il ne peut être que positif comme je vais vous le démontrer dans cette partie.
Je vais donc, pour chaque interface, vous présenter la version de base, les discussions autour de celle-ci avec le client pour enfin arriver à une version finale.

1- Développement des interfaces de gestion des gabarits

- Version 1 :
La gestion des gabarits consiste simplement à ajouter, modifier ou supprimer un gabarit, comme présenté précédemment.

Les interfaces devaient donc permettre la gestion de ces trois actions, j'ai donc créé trois fenêtres différentes et pour les actions « modification » et « suppression », une fenêtre permet de choisir le gabarit souhaité.
Toutes les saisies sont contrôlées, ainsi l'utilisateur ne peut taper que des chiffres dans les cases des dimensions et il ne peut valider un ajout ou une modification d'un gabarit que si toutes les dimensions et le nom sont saisis correctement.

Voici la première version de ces interfaces :

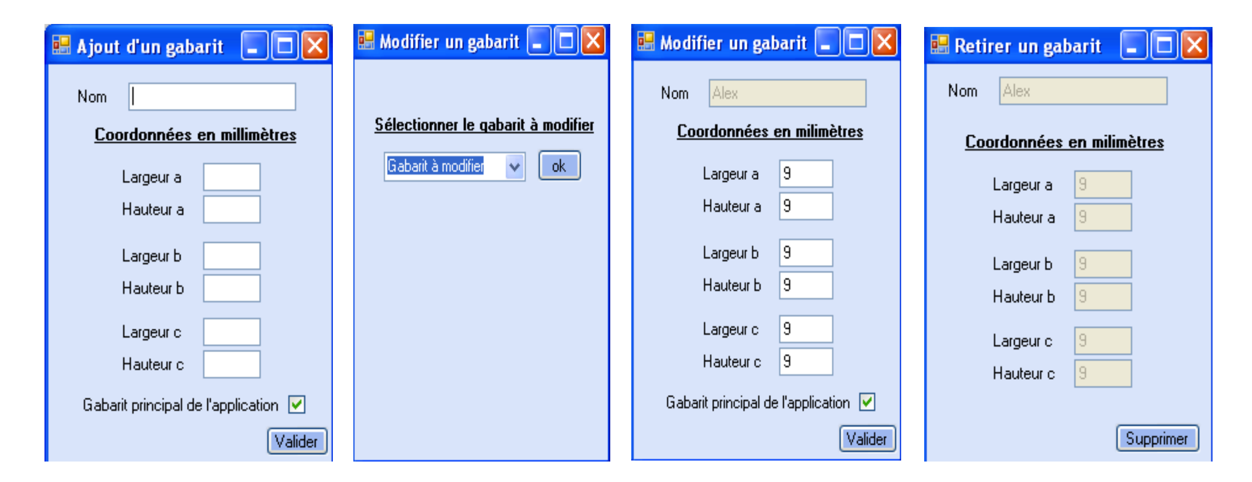


Figure 22 - Interfaces de gestion des gabarits (v1)

- Remarques du client :

A la vue des premières interfaces, Jean-Marc a tout de suite insisté sur le fait que les utilisateurs (les médecins) devaient trouver facilement tout ce qu'ils cherchaient puisqu'ils ne sont pas des

utilisateurs chevronnés de l'informatique. Cette remarque a entraîné la première modification à apporter aux interfaces, parmi d'autres que je citerai juste après :

1) Il faut **ajouter un bouton « sortir » sur toutes les fenêtres** car les utilisateurs préfèrent avoir un bouton intégré dans la fenêtre pour être certains de l'action réalisée, plutôt que de cliquer sur la croix rouge de la fenêtre Windows.

2) Pour la suppression d'un gabarit, l'utilisateur sélectionne un gabarit et clique sur « ok ». L'utilisateur va alors croire que son gabarit va directement être supprimé après le clic sur le bouton alors que ce n'est pas le cas. L'idéal, pour rendre cette gestion des gabarits intuitive serait de **rassembler toutes ces actions dans une seule interface.**

3) Le texte indiquant que les dimensions sont en millimètres ne sera jamais lu par l'utilisateur qui va en premier se focaliser sur les dimensions à saisir. Il faut alors **indiquer « mm » derrière chacune des cases des différentes dimensions.**

4) **Intégrer le schéma représentatif d'un gabarit dans cette interface** semble important pour l'utilisateur qui pourra alors se repérer dans le schéma pour saisir les dimensions.

5) La dernière remarque du client est arrivée après cette discussion et personne n'y avait pensé auparavant : **ajouter une fonctionnalité qui permettra de sauvegarder un gabarit** sur l'ordinateur, pour qu'il soit réutilisé plus tard lors d'une nouvelle utilisation. La sauvegarde pourra par exemple être réalisée sous forme d'un fichier texte.

- Version 2 :

Excepté la dernière remarque, toutes les remarques du client Jean-Marc ont été prises en compte et m'ont permis de réaliser cette deuxième version de l'interface de gestion des gabarits. Voici donc la nouvelle version de l'interface, plus implicite et plus intuitive pour le client :

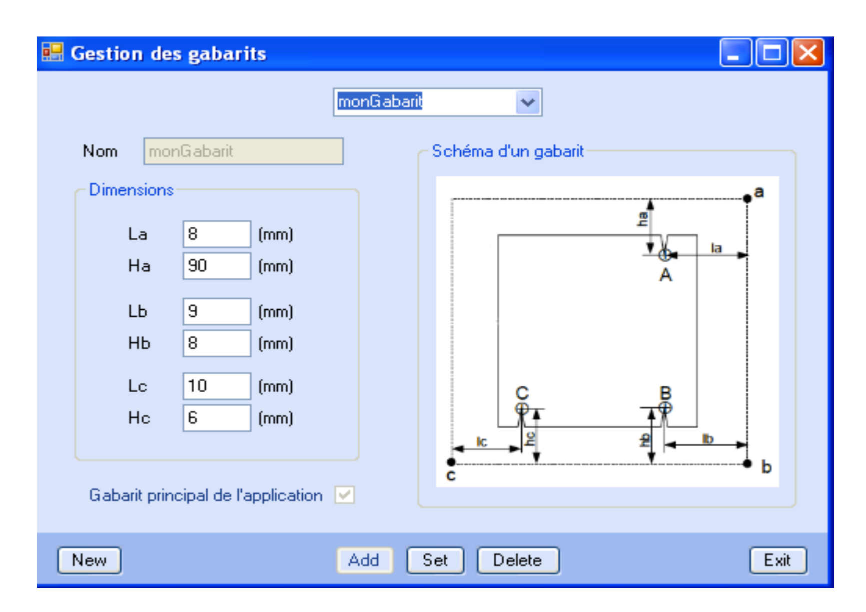


Figure 23 - Interface "Gestion des gabarits" (v2)



2- Développement de l'interface « positionnement des points repères de la photo »

- Version 1 :

La première des choses à réaliser pour cette interface est le choix par l'utilisateur de la photo de la peau, une fois que celle-ci est chargée, il ne reste plus qu'à positionner les points repères sur la photo. Pour cela, l'utilisateur commence par le premier point repère et clique sur l'endroit de la photo pour mettre à jour les coordonnées du premier point. Une fois que l'utilisateur a positionné les trois points repères, un bouton « modifier » apparaît et celui-ci lui permet de revenir sur ces saisies et de les modifier en cliquant sur le texte « point 1 » s'il souhaite modifier le premier point.

J'ai également rajouté la possibilité de modifier les coordonnées saisies grâce aux comboBox des coordonnées x et y qui permettent à l'utilisateur de modifier les coordonnées de un en un, pour gagner en précision.
Une autre fonctionnalité à laquelle j'ai pensé et que j'ai ajouté, est le changement de couleurs des points repères, à la base rouge. Ainsi l'utilisateur peut choisir la couleurs des points parmi une gamme de couleurs basiques (rouge, bleu, vert, blanc, noir, jaune, orange) pour mieux s'adapter aux couleurs de la photo.

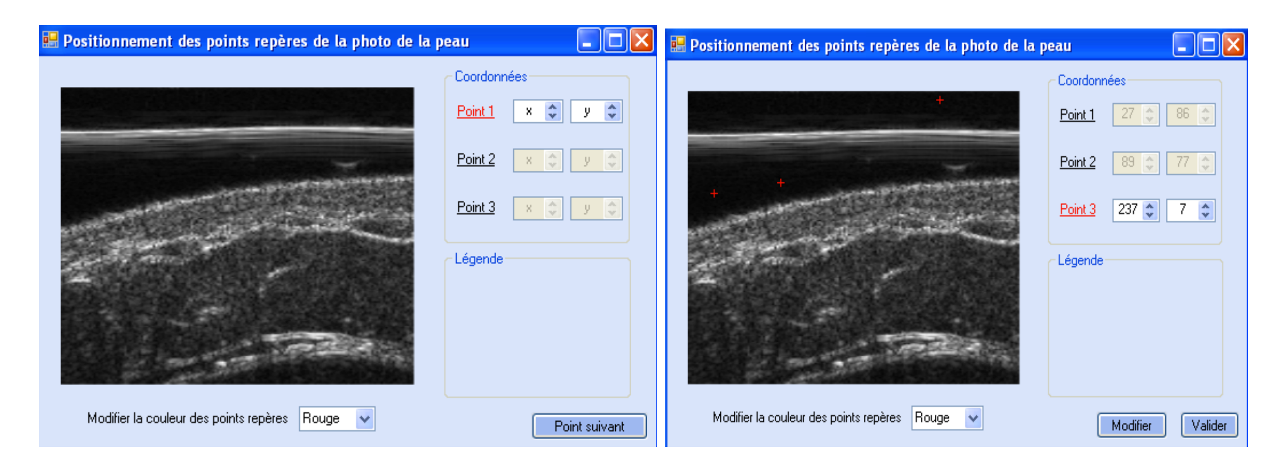


Figure 24 - Interface "Positionnement des points repères de la photo" (v1)

Comme vous pouvez le remarquer sur les interfaces ci-dessus, la partie réservée à la légende n'est pas renseignée. En effet, en réalisant cette interface, je me suis rendu compte que je n'avais pas compris comment cette légende était construite : est-elle construite en même temps que les points saisis ? Avant ? L'utilisateur connaît-il le positionnement des points sur la légende avant d'ouvrir la photo ? Que se passe t'il si la photo a été prise à l'envers ?

C'est grâce à toutes ces questions que nous avons facilement trouvé une solution avec le client pour parvenir à la construction de la légende, comme je vais le décrire dans les remarques du client.

- Remarques du client (Réunion 1):

1) Il ne faut pas attendre d'avoir placer les trois points repères pour pouvoir cliquer sur un point et le modifier. Cette option doit être disponible dès l'ouverture de la fenêtre pour permettre au client de **modifier n'importe quand les coordonnées d'un point.**

2) La première remarque en entraîne une autre : en effet, pour cliquer sur un point et le modifier, l'utilisateur devait cliquer sur le texte « point 1 », ce qui n'est pas très implicite. Il faut donc **remplacer les textes « point i » par des boutons.**

3) Plutôt que d'ouvrir la photo à l'ouverture de l'interface, il serait plus judicieux d'ouvrir la photo de la peau au préalable. Il faut alors **rajouter dans le menu « File » la fonction « OuvrirPhoto ».**

4) Concernant la discussion sur la construction de la légende : celle-ci est connu de l'utilisateur avant qu'il n'ouvre l'interface de positionnement des points repères et elle permettra à l'utilisateur de connaître l'emplacement d'un point repère grâce à la légende. Si l'utilisateur clique sur le point 1, le point 1 de la légende apparaîtra d'une couleur différente des deux autres. Ainsi, le mieux serait de **proposer, dès l'ouverture de la fenêtre, un choix entre quatre légendes** qui correspondent aux quatre différents angles sous lesquels l'utilisateur peut réaliser la photographie. L'utilisateur choisira alors celle qui correspond à l'orientation de la photo.

- Version 2 :



Figure 25 - Interface "Positionnement des points repères de la photo" (v2)

Toutes les modifications ont donc été apportées à l'interface. Sur la gauche, il s'agit du choix qui est proposé à l'utilisateur pour sélectionner la légende correspondant à l'orientation de la photo de la peau. Enfin sur la droite, les textes « Point i » ont bien été remplacés par des boutons et la légende apparaît désormais.



- Remarques du client (Réunion 2):

1) Une fois que les trois points repères sont placés, il est impossible de distinguer quel est le point actif sur la photo car tous les points repères sont dessinés de façon identique. Ainsi, il faudrait **différencier le point actif de ceux non actif sur la photo**.

2) Les points indiqués sur la légende devront correspondre à ceux dessinés sur la photo.

3) **Indiquer « x » et « y » au dessus des comboBox** car lorsque tous les points repères sont positionnés les x et y contenus dans les comboBox ne sont plus visibles.

4) Les futurs utilisateurs de l'application pourraient souhaiter sélectionner de nouveau une légende si jamais leur premier choix n'était pas correct, il faudra donc **ajouter un bouton « suivant » sur le choix des légendes** et un **bouton « précédent » sur l'IHM principale**.

- Version 3 :



Figure 26 - Interface "Positionnement des points repères de la photo" (v3)

Voici donc la dernière version de la fenêtre permettant de positionner les points repères sur la photo de la peau. Plus aucune modification ne sera apportée. Vous pouvez remarquer les différents changements, tels que les légendes « x » et « y » ajoutées au dessus des comboBox ou encore la distinction entre le points repère actif et les points repères non actifs sur la photo et sur la légende (un « X » pour le point repère actif et un « + » pour les points repères non actifs

3- Développement de l'interface « positionnement des points repères de l'échographie »

Les remarques apportées par la client sur cette interface sont les mêmes que pour l'interface « positionnement des points repères de la photo de la peau », je ne reviens donc pas dessus même si je vais rajouter d'autres remarques, spécifiques à cette interface.

- Version 1 :

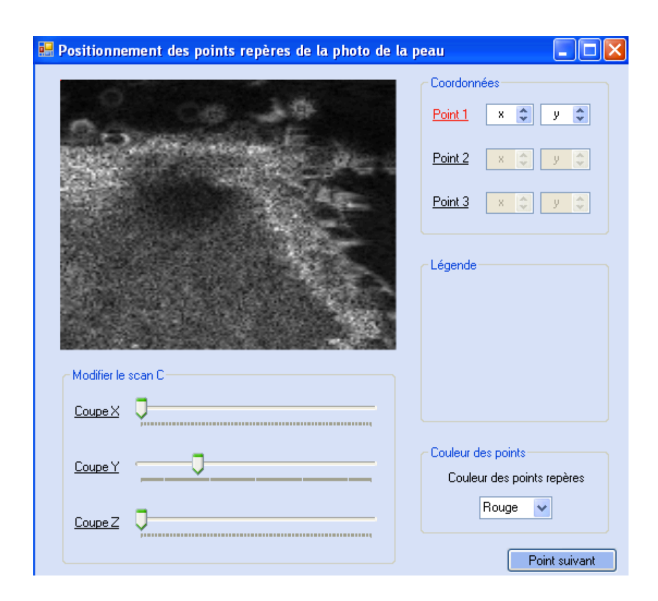


Figure 27 - Interface "Positionnement des points repères de l'échographie" (v1)

- Remarques du client (réunion 1) :

1) La présence des axes X et Y n'est pas nécessaire pour cette interface puisque l'utilisateur souhaite uniquement se balader dans l'axe Z représentant les scans C. **Il faut donc enlever les coupes X et Y.**

2) Il faut pouvoir **sauvegarder la valeur de l'axe Z** qui nous permettra par la suite de définir un nouvel axe si la peau n'est pas parallèle aux scans C.

- Version 2 :

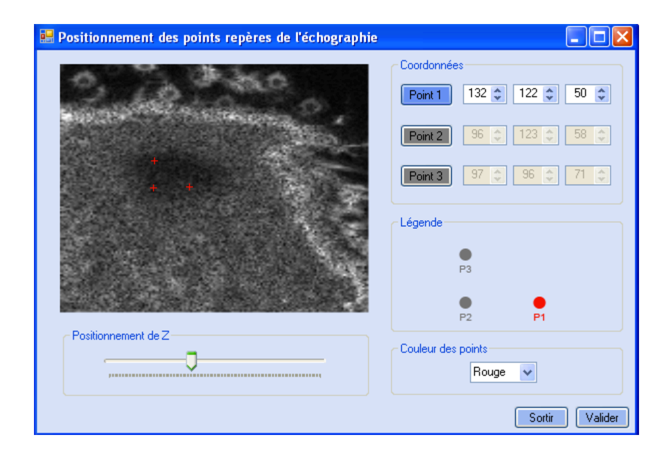


Figure 28 - Interface "Positionnement des points repères de l'échographie" (v2)



Voici donc la seconde version de l'interface « Positionnement des points repères de l'échographie », avec toutes les modifications qui ont été réalisées et dont le fonctionnement est le même que l'interface présentée précédemment.

- Remarques du client (réunion 2) :

1) **Ajout d'une indication sur le numéro de l'image de l'échographie** en plus de la barre déjà existante car le futur utilisateur souhaitera à coup sûr plus de précision que cette barre.

2) Sur la fenêtre permettant de choisir la légende appropriée, il faudra également **ajouter la possibilité de parcourir les scans C (sur le choix des légendes)**.

- Version 3 :

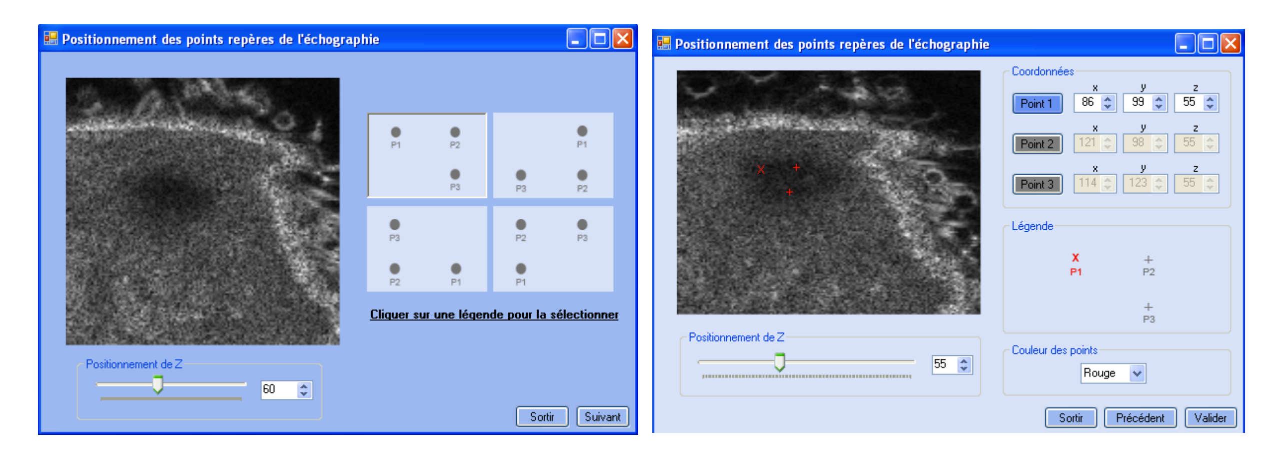


Figure 29 - Interface "Positionnement des points repères de l'échographie" (v3)

4- Développement de l'interface « Fusion »

Cette interface, qui va également englober le module de « fusion » permettant d'obtenir le résultat de la fusion entre l'échographie et la photo de la peau, devra comprendre plusieurs fonctionnalités :

- Une partie de l'interface contenant la photo d'origine avec les points repères placés par l'utilisateur ainsi que les points repères recalés grâce aux translations imposées par le gabarit.
- L'autre partie contiendra la fusion de la photo avec l'échographie de la peau. Il sera possible de se déplacer en profondeur dans l'échographie (parmi les scans C).
- Le fonctionnement de la partie contenant le résultat de la fusion devra permettre à l'utilisateur de modifier la transparence de la photo de la peau ou du scan C affiché. Et tout comme sous Photoshop, l'utilisateur pourra choisir laquelle des deux images (photo de la peau ou échographie) se trouve au premier plan.

- Problème rencontré :
La photo de la peau étant d'une taille beaucoup plus importante que l'échographie, la fusion de la photo avec l'échographie ne donnera assurément pas un bon résultat puisque si la photo est diminuée, alors l'échographie doit également l'être. L'échographie sera alors trop petite et le résultat obtenu sera loin d'être satisfaisant au niveau de la qualité.
C'est pour cela que j'ai proposé une découpe de la photo de la peau de la taille de l'échographie, tout cela autour des points repères positionnés sur la photo. Ainsi, nous obtiendrons un zoom de la photo de la peau, de la taille de l'échographie.

Pour découper la photo, je me base sur la légende, c'est à dire l'angle sous lequel la photographie et l'échographie ont été réalisées. Les points repères de la photo ainsi que de l'échographie sont superposés, les coordonnées du point repère 1 de l'échographie vont donc me permettre de trouver le coin haut gauche de la future image. Une fois que ce point est trouvé, il suffit de parcourir tous les points de la photo depuis ce coin gauche jusqu'au coin en bas à droite donnant une nouvelle image de la même taille que l'échographie.

Voici un algorithme et un schéma détaillé permettant de résumer ces explications :

```
1- Recherche le point repère 1 (p1) de l'échographie
2- Rechercher le coin haut gauche de la future image

    - xCoinGauche = xPhoto - xEcho
    - yCoinGauche = yPhoto - yEcho

3- Parcours des points de la nouvelle image (de la taille de l'échographie)
pour l'enregistrer

    Pour i de 0 à largeurEchographie Faire
        Pour j de 0 à hauteurEchographie Faire

            newImage[i][j] = photo[]xCoinGauche+i[yCoinGauche+j]

        FinPour
 FinPour
```

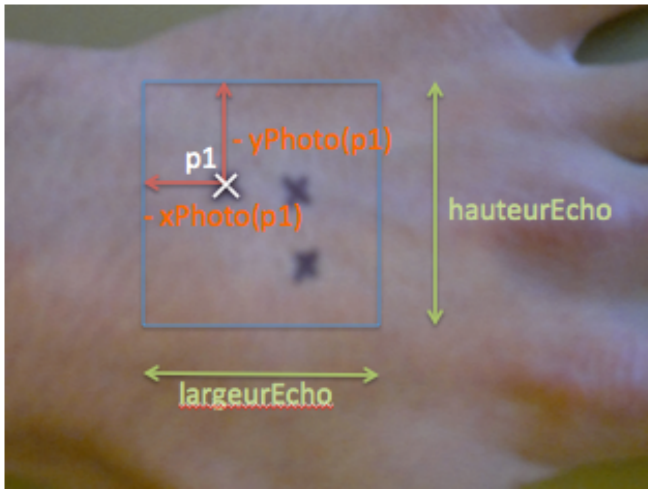


Figure 30 - Découpage de la photo de la peau



Version 1 :

Figure 31 - Interface "Fusion"

Sur la partie gauche de l'interface, il s'agit de la photo sous sa forme d'origine, avec la possibilité d'afficher les points repères positionnés par l'utilisateur ainsi que les points repères repositionnés à l'aide du gabarit. Cela permet à l'utilisateur de pouvoir se repérer sur la photo puisque certains indices comme les veines ou les muscles peuvent lui être utiles pour se repérer facilement.

La partie de droite concerne directement le résultat de la fusion, avec un mode de fonctionnement proche de Photoshop puisqu'il est possible de choisir quelle image sera en premier plan, puis il est également possible de faire varier la transparence de la photo de la peau ainsi que de l'échographie.

Une fois que l'utilisateur a trouvé la fusion parfaite, il peut sauvegarder le résultat de la fusion dans le menu « Fichier/Sauvegarder fusion ».

Améliorations à apporter :
L'affichage de la fusion n'est pas totalement instantané lorsque l'utilisateur déplace le curseur permettant de modifier la transparence de la photo de la peau ou de l'échographie. Cela vient du fait que les pixels sont recalculés à chaque modification du curseur et ce calcul prend du temps et des ressources.

J'ai bien tenté de sauvegarder tous les pixels dans des tableaux de couleurs, mais l'application refusait de se lancer, sûrement dû aux trop nombreux calculs demandés par cette opération. En

effet, la transparence possède 150 valeurs, et une image est en moyenne de 150*150 pixels (beaucoup plus pour la photo de la peau). Il faudra donc par la suite trouver un moyen de ne pas avoir à recalculer tous les pixels lors de la modification de la transparence de la photo ou de l'échographie.

5- Autres remarques du client

Toutes ces remarques ont influencé un changement du menu pour que celui-ci devienne également plus implicite. Le menu gabarit sera alors remplacé par « Configuration » avec la fonction « Gabarit » qui permettra de gérer les gabarits. Pourquoi ? Puisque ce menu sera amener à évoluer avec l'ajout de nouvelles fonctions de configurations. De même pour le menu « fusion » dont les fonctions seront renommées « PlacerReperesEchographie » et « PlacerReperesPhoto ».

Une autre remarque très pertinente concernant l'utilisation de l'application et à laquelle nous n'avions pas pensé jusqu'ici est la sauvegarde des fusions réalisées dans les précédentes utilisations et qui pourraient être importées dans l'application pour que l'utilisateur puisse reprendre un travail en cours. Par exemple, le praticien prépare préalablement une opération à l'aide de l'application. La fusion est réalisée et le jour de l'opération il n'a plus qu'à importer ce travail dans l'application qui affichera directement le résultat. Pour cela, il faudra sauvegarder les points repères, les chemins de l'acquisition 3D, de la photo de la peau.

CONCLUSION SUR LE DEVELOPPEMENT DES INTERFACES

Encore une fois, j'ai pu constaté à travers cette partie « développement des interfaces » qu'un contact permanent avec le client permet de progresser très rapidement et dans le bon sens. En effet, les différentes rencontres avec le client après chaque progression des interfaces a permis de garantir la livraison de l'outil exigé par le client. C'est donc beaucoup de temps de gagner pour la fin du projet et le déploiement et la formation sur le logiciel puisque le client connaît parfaitement les exigences des futurs utilisateurs.



PARTIE 4 : RECALAGE DU VOLUME D'ACQUISITION

Il ne me restait que très peu de temps pour réaliser cette partie et je n'ai pas eu le temps de totalement la terminer même si une proportion importante du travail a été effectuée.

Dans le travail que j'ai réalisé, je ne m'occupe pas de la détection de la surface de la peau, mais je tente uniquement de réaliser une rotation de l'échographie 3D en lui indiquant un angle de rotation manuellement. J'ai d'abord pensé réaliser cette rotation sur l'ensemble de l'acquisition 3D, ce qui inclut donc une rotation de chaque point 3D (x, y et z) de l'échographie.

Fonction RotateImage3D():

```
void Image3D::RotateImage3D()
{
    int centerX, centerY, centerZ, newX, newY, newZ;
    array<Color, 3>^ imageRotate = gcnew array<Color, 3>(largeur, hauteur, profondeur);

    // centre du cube
    centerX = largeur/2;
    centerY = hauteur/2;
    centerZ = profondeur/2;

    double angle = Math::PI*(15)/180;

    // initialisation des pixels de la matrice en rouge
    for(int x=0; x<largeur; x++)
    {
        for(int y=0; y<hauteur; y++)
        {
            for(int z=0; z<profondeur; z++)
            {
                // calcul des nouvelles coordonnées du pixel
                newX = 1*(x-centerX)+centerX;
                newY = cos(angle)*(y-centerY)-sin(angle)*(z-centerZ)+centerY;
                newZ = sin(angle)*(y-centerY)+cos(angle)*(z-centerZ)+centerZ;

                if((newX>-1 && newX<largeur) && (newY>-1 && newY<hauteur) && (newZ>-1 && newZ<profondeur))
                {
                    imageRotate[newX, newY, newZ] = colorImage[x, y, z];
                }
            }
        }
    }

    colorImage = imageRotate;
}
```

Chaque point de l'échographie est donc parcouru et ses nouvelles coordonnées sont alors calculées avec les formules spécifiques à une rotation 3D, soit les suivantes :

- Pour la coordonnée X

  newCoordX = 1(x-centreX)+centreX*

- Pour la coordonnée Y

  newCoordY = cos(angle)(y-centreY)-sin(angle)*(z-centreZ)+centreY*

- Pour la coordonnée Z

  newCoordZ = sin(angle)(y-centreY)+cos(angle)*(z-centreZ)+centreZ*

Le nouveau point (newCoordX, newCoordY, newCoordZ) va alors prendre le pixel de l'ancien point (x, y, z). Une fois que tous les points ont été parcourus, cela nous donne une nouvelle échographie 3D, qui a subi une rotation de 15 degrés.

Résultat obtenu :

La rotation de 15 degrés a bien été réalisée sur l'échographie, mais le résultat obtenu n'est pas satisfaisant puisque des traits blancs apparaissent sur l'échographie. Ces traits sont réguliers et pourraient provenir d'une erreur dans le parcours des points, mais je n'ai pas eu le temps d'approfondir les recherches pour trouver la source de cette erreur.

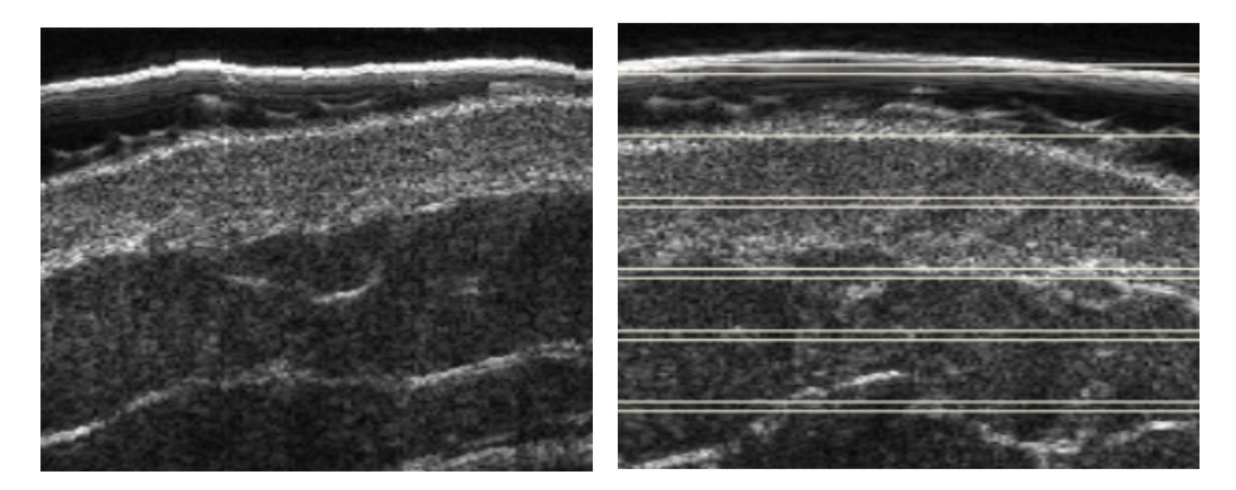


Figure 32 - Rotation 3D de l'échographie (à gauche = avant, à droite = après)

Fonction RotationImage2D() :

J'ai alors tenté une seconde façon de réaliser cette rotation, c'est à dire une rotation 2D sur un seul axe en modifiant les points x et y de chaque image de l'axe, ce qui changera obligatoirement les autres axes étant donné qu'ils partagent tous les même points. Néanmoins cette solution n'a pas changé le résultat qui est resté exactement le même que ci-dessus.

```
void Image3D::RotateImage2D()
{
    int centerX, centerY, newX, newY;
    array<Color, 3>^ imageRotate = gcnew array<Color, 3>(largeur, hauteur, profondeur);

    double angle = Math::PI*(15)/180;

    for(int z=0; z<profondeur; z++)
    {
        for(int x=0; x<largeur; x++)
        {
            for(int y=0; y<hauteur; y++)
            {
                // calcul des nouvelles coordonnées du pixel
                newX = cos(angle)*(x-centerX)-sin(angle)*(y-centerY)+centerX;
                newY = sin(angle)*(x-centerX)+cos(angle)*(y-centerY)+centerY;

                if((newX>-1 && newX<largeur) && (newY>-1 && newY<hauteur))
                {
                    imageRotate[newX, newY, z] = colorImage[x, y, z];
                }
            }
        }
    }

    colorImage = imageRotate;
}
```



PARTIE 5 : EXPORTER ET IMPORTER UN GABARIT ET UN PROJET

Jusque là, l'application développée ne permet pas de sauvegarder le travail réalisé par l'utilisateur, ainsi, dès que celui-ci quitte l'application, il ne peut retrouver ce qu'il a fait. En effet, comme expliqué précédemment, le praticien ne réalisera pas ce travail le jour de l'opération et encore moins juste avant ou pendant l'opération. C'est un travail qui sera réalisé au préalable et qui devra donc être chargé le jour de l'opération. C'est donc dans cette optique que la sauvegarde d'un projet semble indispensable au sein de l'application.

De plus, les gabarits utilisés seront souvent les même, il est donc également intéressant de pouvoir les sauvegarder afin de les importer facilement lors de futures utilisations.

1- Exporter et importer un gabarit

Ces deux options ont été ajoutées à l'interface « Gestion des gabarits », ainsi l'utilisateur peut importer et exporter les gabarits, selon son besoin.

La sauvegarde d'un gabarit est réalisée dans un fichier texte et contient toutes les informations nécessaires, c'est à dire le nom ainsi que les dimensions. Pour importer un gabarit, l'utilisateur doit rechercher ce fichier de sauvegarde et le gabarit sera alors chargé en mémoire dans l'application.

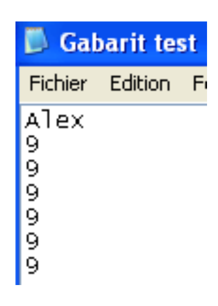


Figure 33 - Fichier de sauvegarde d'un gabarit

Améliorations à apporter :

Très peu de vérifications sont réalisées sur un fichier importé, il faudrait par exemple vérifier l'intégrité des données contenues dans le fichier. Pour l'instant les seules vérifications sont les suivantes :

- Si le nombre de lignes correspond exactement à ce qui est attendu, dans le cas d'un gabarit, il s'agit de 7 lignes : une pour le nom du gabarit ainsi que ses 6 dimensions. S'il manque une ligne ou s'il y en a plusieurs en trop, le gabarit ne sera pas importé.

- Si le gabarit existe déjà, celui-ci ne sera pas non plus importé.

Les vérifications à ajouter seraient donc le test de l'intégrité des données, c'est à dire vérifier que les dimensions sont bien des entiers par exemple. De plus, il faudrait verrouiller le fichier créé pour empêcher toute modification d'un utilisateur malveillant.

2- Exporter et importer un projet

Le menu « Importer un projet » est accessible dès le lancement de l'application tandis que le menu « Exporter un projet » n'est accessible que lorsque l'utilisateur a réalisé toutes les manipulations nécessaires pour arriver au résultat de la fusion. Ainsi, l'utilisateur doit créer ou importer un gabarit, positionner les points repères sur l'échographie et sur la photo de la peau avant de pouvoir exporter ce travail. Si l'une de ces trois étapes n'est pas réalisée, l'utilisateur ne pourra pas sauvegarder son travail.

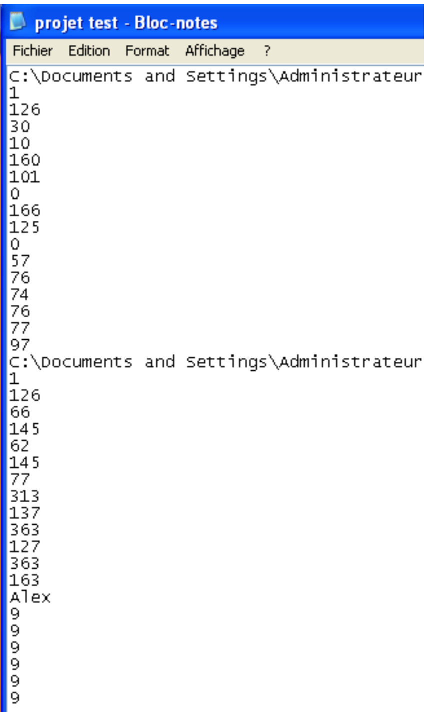
Lorsque l'utilisateur souhaite importer un projet, il doit simplement retrouver le fichier de sauvegarde qui contient toutes les données sur le travail préalablement réalisé. Une fois le fichier lancé depuis l'application, tous les modules sont chargés et toutes les interfaces sont comme si l'utilisateur n'avait jamais quitté l'application.

Concernant la sauvegarde en elle-même, le fonctionnement est identique à la sauvegarde d'un gabarit, c'est à dire que les données sont enregistrées dans un fichier texte. Dans l'ordre, les données sauvegardées sont :

- Chemin du dossier contenant les images de l'échographie
- Légende (orientation) de l'échographie
- Points repères de l'échographie, placés sur la pictureBox
- Points repères de l'échographie (calculés avec les ratios)

- Chemin du fichier de la photo de la peau
- Légende (orientation) de la photographie
- Points repères de la photo, placés sur la pictureBox
- Points repères de la photo (calculés avec les ratios)

- Nom du gabarit actif (celui utilisé pour la fusion)
- Dimensions du gabarit


projet test - Bloc-notes

Fichier Edition Format Affichage ?

```
C:\Documents and Settings\Administrateur
1
126
30
10
160
101
0
166
125
0
57
76
74
76
77
97
C:\Documents and Settings\Administrateur
1
126
66
145
62
145
77
313
137
363
127
363
163
Alex
9
9
9
9
9
9
```

Figure 34 - Fichier de sauvegarde d'un projet

PARTIE 6 : LES AMELIORATIONS A APPORTER

Le projet arrivant à sa fin, une grande partie des modules souhaités par le client a été implémentée et certains d'entre eux devront être améliorés pour compléter l'application et rendre son utilisation plus agréable.

C'est le cas du module de recalage de l'échographie 3D, qui n'a pas été achevé et qui est une fonctionnalité souhaitée par l'utilisateur étant donné la complexité de réaliser une échographie parfaitement parallèle à la surface de la peau sur certaines parties du corps humain. Pour ce module, la rotation souhaitée se réalise correctement mais le résultat obtenu n'est pas satisfaisant car la nouvelle acquisition, contient des traits blancs. Il faudra donc soit repartir de ce qui existe et comprendre d'où proviennent ces traits blancs, soit reprendre ce module à zéro et peut être directement intégrer une gestion automatique du recalage au chargement de l'échographie.

Le second module a amélioré est celui de la fusion de la photo de la peau avec un scan C de l'échographie 3D. En effet, une latence apparaît lorsque l'on souhaite modifier la valeur de la transparence du scan C ou de la photo de la peau. Cela vient du fait que les valeurs des pixels sont recalculées à chaque modification de la valeur de la transparence. J'avais au préalable tenter d'enregistrer en mémoire ces couleurs pour ne pas les recalculer à chaque mouvement, mais leur place en mémoire était trop importante et l'application refusait de se lancer.

D'autres améliorations apparaîtront probablement lorsque le matériel pour réaliser les tests sera disponible, il faudra alors adapter l'application aux nouvelles contraintes, s'il y en a, ou simplement apporter une correction à certains éléments si des choses auxquelles nous n'avions pas pensé ressortent avec les tests.

BILAN - LA GESTION DU PLANNING

Le projet aurait pu se dérouler selon le premier planning établi, c'est à dire une première période de 2 mois pour se familiariser avec le sujet, le cahier des charges, pour réaliser le cahier de spécifications système et enfin pour ressortir le module d'affichage de l'application existante. Néanmoins, ce dernier point a posé plus de souci que prévu et a par conséquent décalé toute la suite du projet.

Comme vous pouvez le voir sur le premier planning, qui était prévisionnel, la conception et le développement de l'application devaient débuter en décembre 2011.

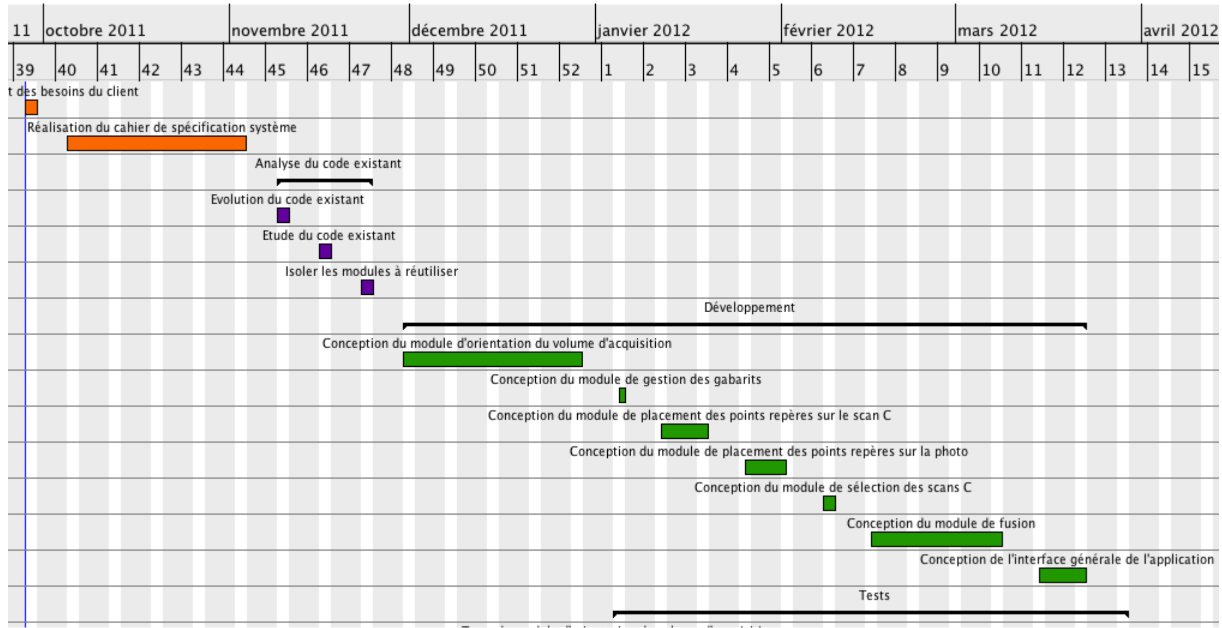


Figure 35 - Planning prévisionnel

Mais avec le retard engendré par l'étude du code existant, tout le reste du projet a été décalé et ce qui est devenue la partie la plus importante du projet, la modélisation de l'application a finalement pris deux mois sur l'ensemble du projet, réduisant la partie développement à seulement trois mois.

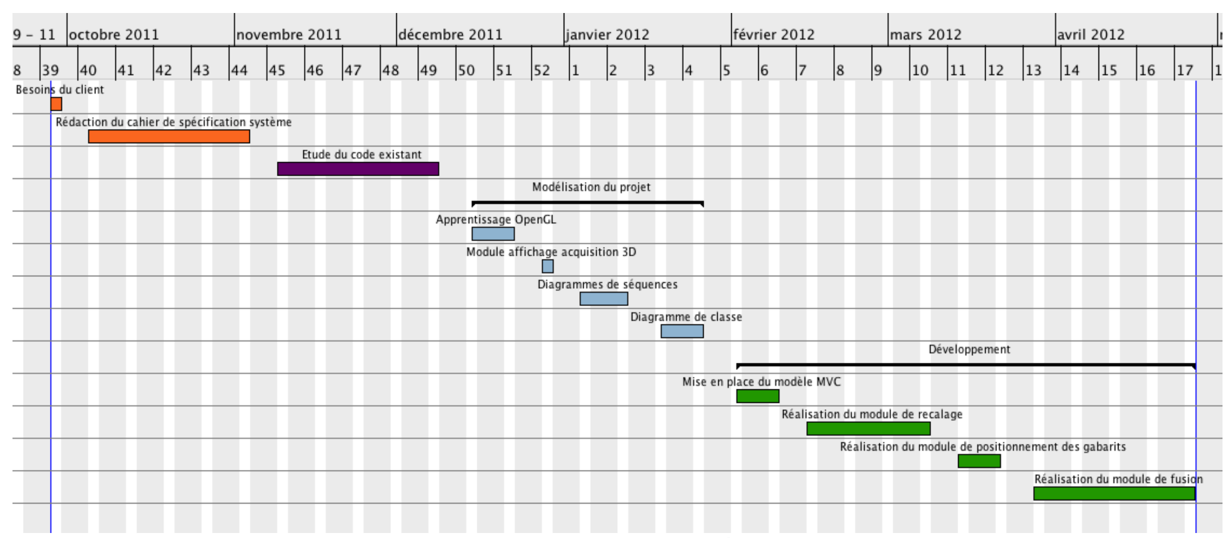


Figure 36 - Planning réel



CONCLUSION

Dans ce rapport, j'ai repris toutes les étapes qui m'ont permis de mener à bien ce projet de fin d'études. De la découverte du sujet jusqu'au développement de celui-ci, tout y est détaillé. L'objectif principal de ce projet était de réaliser une application médicale permettant de faciliter la vision sous cutanée d'une lésion, tout cela à partir d'une application existante qui permettait déjà d'afficher l'aperçu de l'échographie 3D.

J'ai consacré la première partie de ce projet à comprendre et à reprendre la partie affichage 3D de l'application existante pour redémarrer un nouveau projet utilisant cet affichage. La modélisation utilisée n'étant que très peu claire, rendant le module d'affichage difficile à isoler, cela a été une partie compliquée à gérer mais qui m'a également permis de me rendre compte de l'importance d'un projet bien structuré. La modélisation de ce projet est alors devenu le point le plus important, afin que quiconque reprendra la suite du projet puisse facilement comprendre la façon dont j'ai conçu l'application. Je pense avoir réalisé correctement ce point important du projet, notamment grâce aux nombreux diagrammes de séquences et de classes qui illustrent parfaitement toute la structure de l'application.

Ensuite, le développement de l'application a été plus rapide que ce que je n'avais initialement imaginé. De nombreuses rencontres avec le client ont été nécessaires pour réaliser toutes les interfaces de l'application de façon à ce qu'elles répondent parfaitement aux attentes des futurs utilisateurs. Cette partie du projet m'a confirmé l'importance des rapports avec le client pour mener à bien un projet et pour perdre le moins de temps possible.

Et pour faciliter les premières utilisations des clients sur l'application, j'ai réalisé deux versions d'un manuel d'utilisateur, une en français et l'autre en anglais. Ce manuel permet de décrire et d'illustrer toutes les étapes par lesquelles l'utilisateur doit passer pour obtenir le résultat de la photo de la peau fusionnée avec un scan C de l'échographie 3D.

Bien que l'application ne soit pas aujourd'hui totalement terminée, ce projet aura été un enrichissement certain dans ma formation puisque j'ai eu la chance d'aborder tous les points d'un projet, notamment celui de la modélisation, dont le rôle dans un projet de qualité est primordial. J'ai également eu le privilège de découvrir un autre domaine que l'informatique, la dermatologie, et c'est pour moi exactement ce que j'attends de l'informatique pour mon futur, la découverte de nouveaux domaines à travers l'informatique.



ANNEXE 1 – CONFIGURATION VISUAL STUDIO 2010

1- Création d'un nouveau projet « Application console Win32 » sous Visual Studio 2010

Lorsque l'assistant Application Win32 s'affiche :

- cliquer sur précédent
- sélectionner « Application Windows »
- sélectionner « projet vide »
- cliquer sur Terminer

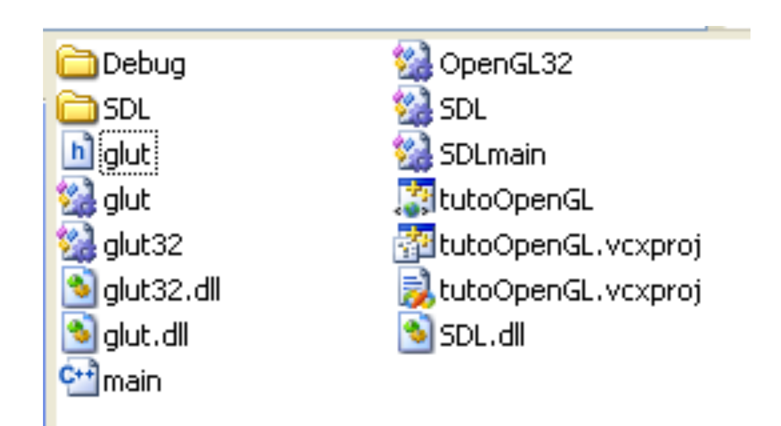
2- Dans le dossier du projet, y ajouter les DLL et librairies nécessaires pour pouvoir coder et générer sans erreurs notre projet :

LIBRAIRIES :

- glut
- glut32
- openGL32
- SDL
- SDLmain

DLL :

- glut32.dll
- glut.dll
- DSL.dll



3- Dans Visual Studio 2010, il faut également ajouter ces liens pour le projet :

- clique droit sur le projet
- Propriétés de configuration/Répertoires VC++/Répertoires Include
- Ajouter le dossier ../monProjet/SDL/include/

- Propriétés de configuration/éditeur de liens/entrée
- Dans « dépendances supplémentaires », ajouter les librairies suivantes : opengl32.lib, glut32.lib, SDL.lib, SDLmain.lib.

- Appliquer tous les changements

4- Ajouter le fichier glut.h dans le dossier monProjet/SDL/include/

5- clic droit propriété projet -> Propriété de configurations/général
« prise en charge du common Language Runtime » = /clr



ANNEXE 2 : MANUEL D'UTILISATION DE L'APPLICATION

I- Chargement en mémoire de l'échographie et de la photo de la peau

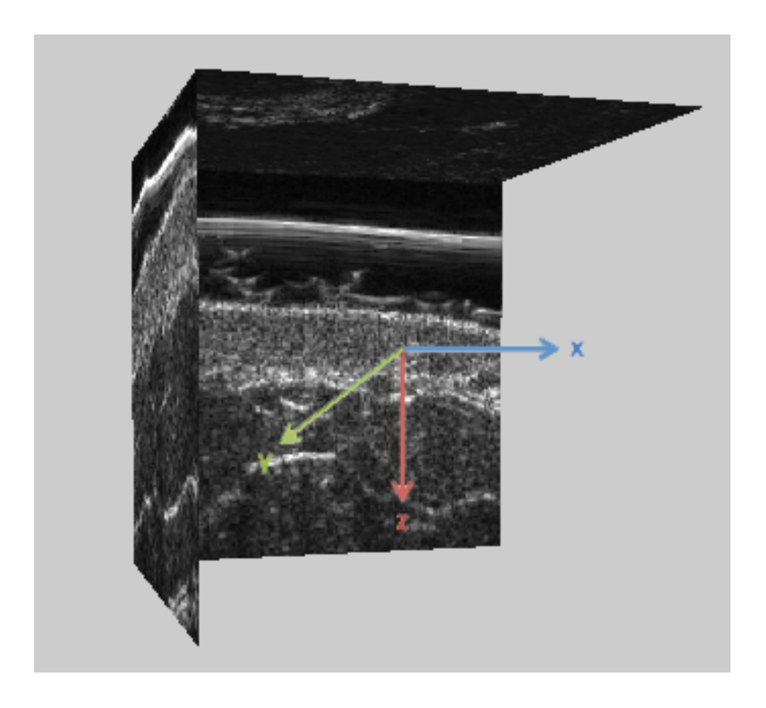


Figure 37 - Menu "Fichier"

A- L'échographie

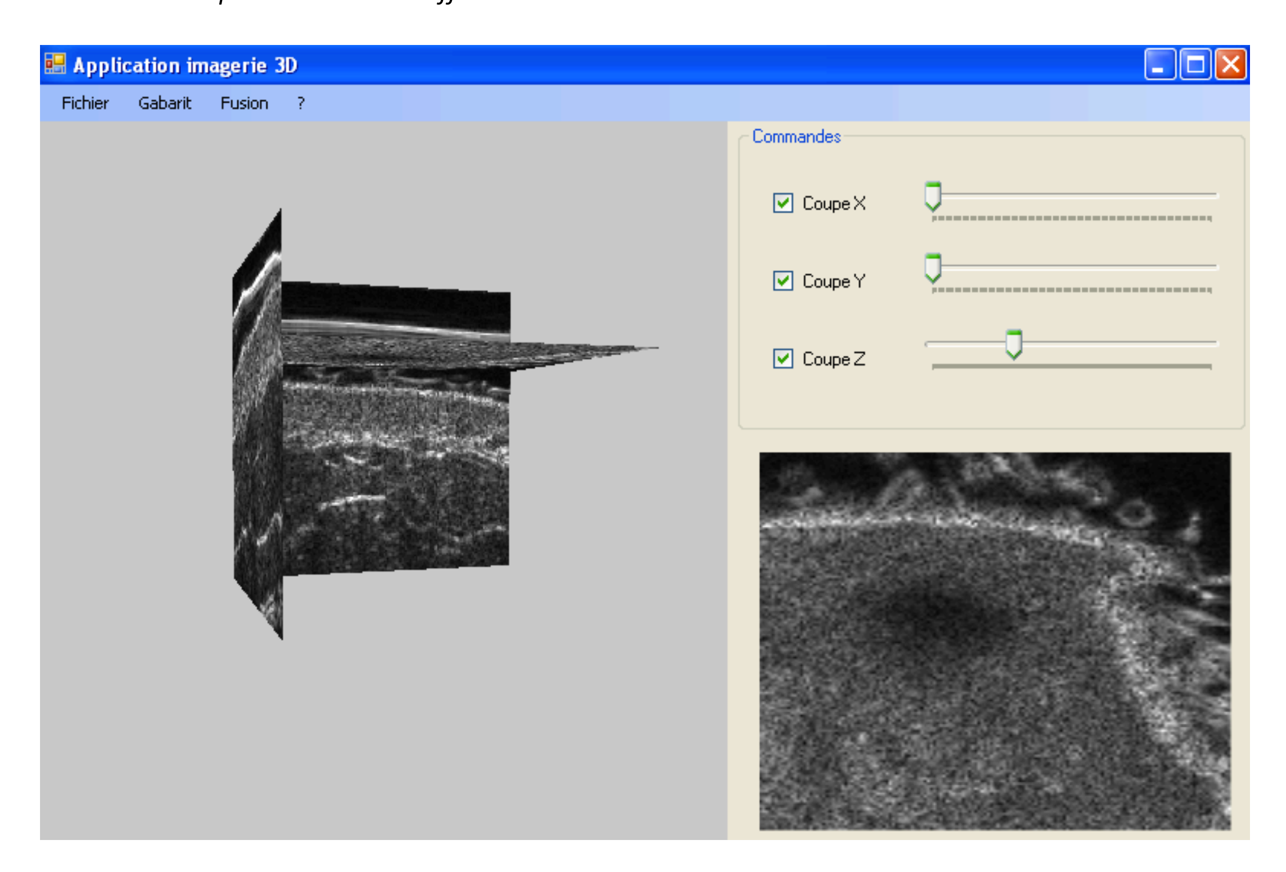
Deux possibilités pour charger l'échographie, soit à partir de du menu « Fichier / Ouvrir Echographie », pour lequel il faudra choisir le dossier contenant toutes les images Bitmap ; soit à partir du menu « Fichier / Ouvrir ATYS », pour lequel il faudra sélectionner un fichier de type Atys.

Une fois l'échographie 3D chargée, il est possible de la visualiser sur la fenêtre principale. Pour parcourir les différents axes de l'échographie 3D, il suffit de cocher les cases correspondant aux axes et de faire varier le Scrollbar d'un des axes à droite.



Concernant l'affichage de l'acquisition 3D, les axes correspondent à ceux indiqués sur le schéma de gauche.

Ex : Sur l'image ci-dessous, l'axe Z est coché en dernier, ce qui va vous permettre de parcourir tous les scans C de l'acquisition et de les afficher en bas à droite.



Remarque : L'échographie ayant été chargée en mémoire, le menu « Fusion / GabaritAcquisition3D » s'est débloqué et est désormais accessible.

B – La photo de la peau

Le chargement de la photo se fait à partir du menu « Fichier / ouvrir photo ». Il suffit simplement de rechercher la photo de la peau dans vos documents et celle-ci sera alors chargée en mémoire.

Cela aura pour conséquence le débloquement du menu « Fusion / GabaritPhotoPeau ».

II- Création d'un gabarit



Figure 38 - Menu "Gabarit"

L'ouverture du menu « Gabarit / Gestion » va vous ouvrir l'interface permettant de gérer tous les gabarits. Depuis celle-ci, il sera possible de créer, modifier, supprimer, exporter et importer un gabarit.



Lors de la première ouverture de cette interface, aucun gabarit n'a été créé, il faut donc soit en importer un soit en créer un. Nous allons en créer un en cliquant sur le bouton « new », ce qui va vous afficher ceci :

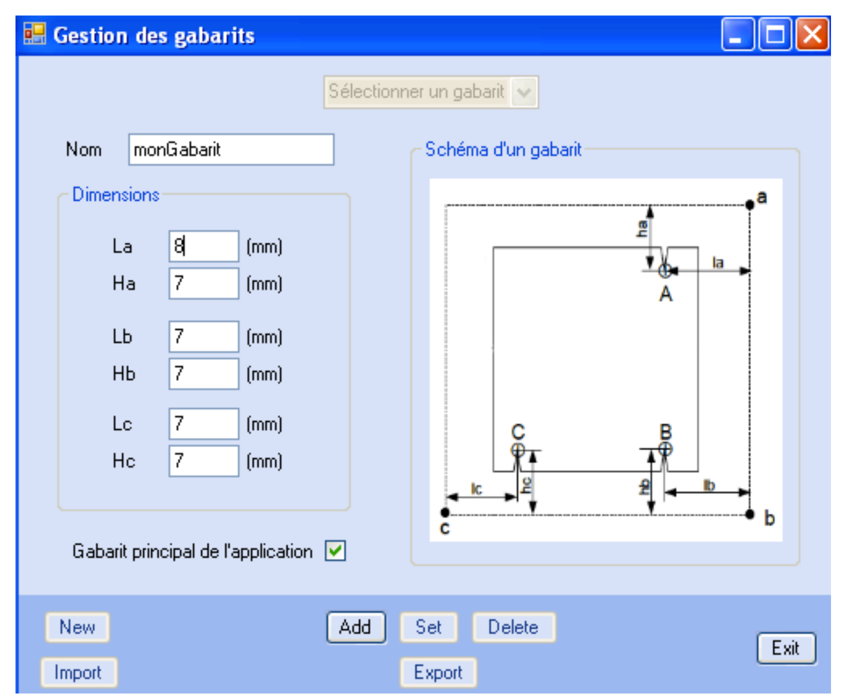


Figure 39 - Interface "Ajouter un gabarit"

Ensuite, il suffit de cliquer sur le bouton « Add » ce qui ajoutera le gabarit en mémoire.
Pour modifier ce gabarit, il faudra le sélectionner, modifier les valeurs et cliquer sur « set » pour valider la modification.

Concernant le schéma détaillant les dimensions d'un gabarit :

- les lettres en majuscules, soit A, B et C correspondent aux points ultrasonores de l'échographie
- les lettres en minuscules, soit a, b et c correspondent aux points repères de la photo de la peau.

Il existe donc un décallage entre les points repères de la photo de la peau et les points repères ultrasonores de l'échographie et ces translations sont représentées par ha, la, …

III- Exporter et importer un gabarit

Pour sauvegarder un gabarit sur votre ordinateur, rien de plus simple, une fois que vous avez créé votre gabarit, cliquez sur « Exporter » et enregistrez le gabarit.

Concernant l'importation d'un gabarit, il faut cliquer sur le bouton « Importer » dans le menu de gestion des gabarits, sélectionnez ensuite le fichier de sauvegarde d'un gabarit et il sera alors chargé en mémoire. Si le gabarit chargé existe déjà, alors il ne sera pas ajouté à la liste des gabarits.

IV- Positionner les points repères de la photo

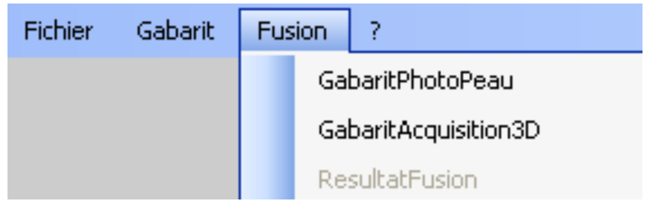


Figure 40 - Menu "Fusion"

Une fois que vous avez chargé la photo de la peau en mémoire, le menu « Fusion / GabaritPhotoPeau » devient accessible et vous pouvez alors positionner vos points repères sur la photo de la peau.

La première étape est de choisir l'orientation avec laquelle vous avez pris votre photo. Quatre choix sont possibles, à vous de sélectionner la légende qui correspond. Pour cela, il vous suffit de cliquer directement sur la légende correspondante et de valider votre choix en cliquant sur « suite ».

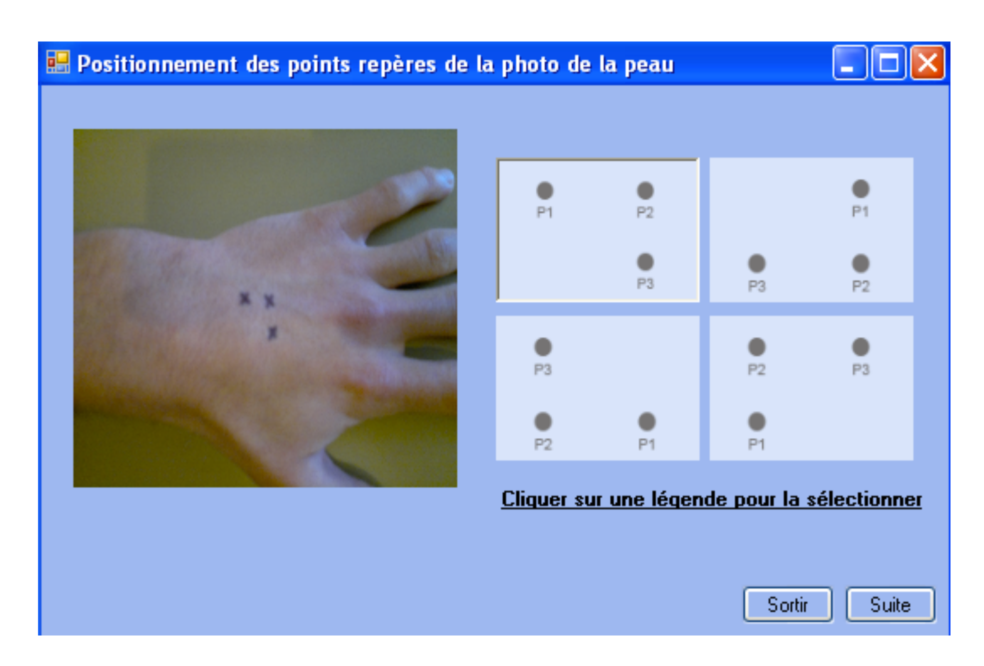


Figure 41 - Interface "Choix légende photo"

Dans notre exemple, nous avons choisi la légende 1.
La seconde interface entraîne la seconde étape qui est le positionnement des points repères de la photo de la peau.



Avant d'expliquer comment placer les points repères sur la photo, nous allons d'abord détailler le rôle de chacun des boutons de l'interface :

- « Point i » : Permet d'activer le point i et donc un clic sur l'image provoquera l'ajout d'un point repère pour le point i. Sachant que i va de 1 à 3.

- « Sortir » :

 - Si le positionnement des points repères n'a jamais été validé, alors l'interface se ferme et rien n'est enregistré.

 - Si le positionnement des points repères a déjà été validé, alors le bouton ne fait que fermer la fenêtre et toutes les données sont enregistrées, ce qui vous permettra de retrouver tout votre travail lors d'une réouverture de la fenêtre.

- « Précédent » : Permet de revenir à la fenêtre de choix de la légende si jamais vous vous rendez compte que votre premier choix n'était pas le bon.

- « Valider » : Valide et enregistre les points repères que vous avez positionnés. SI jamais vous avez oublié un point repère, un message d'avertissement apparaîtra et vous serez obligé de placer ce point. Vous devez donc avoir placer TOUS les points repères pour pouvoir valider et les enregistrer.

- « Légende » : qui n'est pas un bouton mais simplement une indication de quel point est actif et donc quel point va être modifier si vous cliquez sur la photo.

- « Modifier la couleur des points repères » : comme son nom l'indique, permet de modifier les points repères que vous avez positionné sur la photo de la peau.

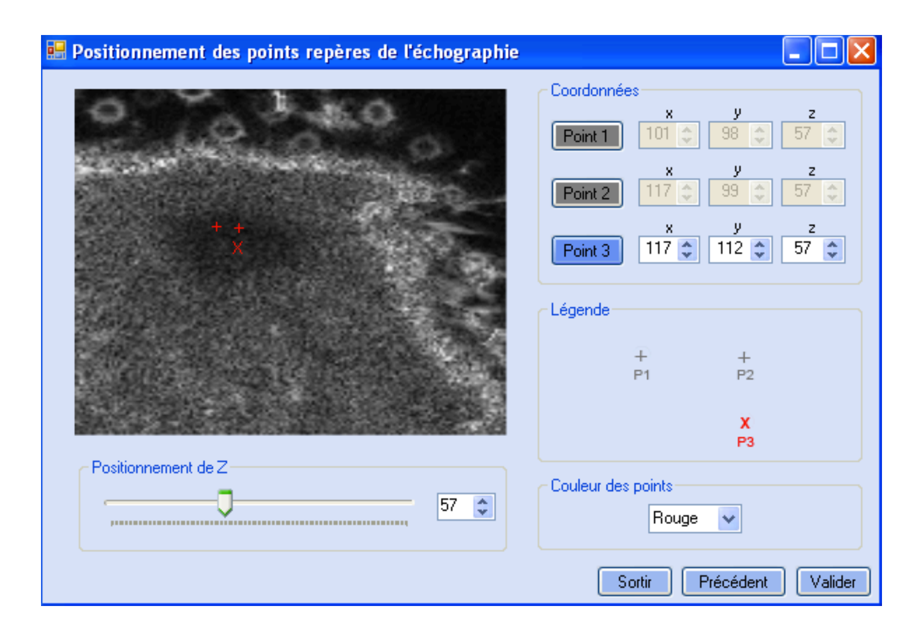
Vous allez désormais positionner vos points repères sur la photo de la peau. Pour cela, si vous souhaitez commencer par le point 1, vous n'avez qu'à cliquer sur l'image et votre point apparaîtra. Les coordonnées exactes seront alors mises à jour dans le cadre « Coordonnées ». Si vous souhaitez positionner un autre point repère, le 2 ou le 3, alors il faut cliquer sur le bouton correspondant et ensuite cliquer sur l'image.



Si vous souhaitez modifier avec précision les coordonnées de vos points repères, vous pouvez modifier la valeur de x ou de y grâce aux deux comboBox dans « Coordonnées ». La modification d'une valeur entraînera le déplacement du point sur l'image.

Cliquez ensuite sur « Valider » pour sauvegarder vos points repères.

V- Positionner les points repères de l'échographie



Le positionnement des points repères de l'échographie se réalise de la même manière que pour les points repères de la photo de la peau.



La seule différence est que vous travaillez sur l’échographie 3D et donc que vous pouvez modifier le scan C affiché sur l’interface en modifiant le positionnement de Z. Cela vous permet de mieux visualiser certains points repères si jamais ils ne sont pas sur le même niveau, ce qui est fort probable.

VI- Afficher le résultat de la fusion

Une fois que les points repères ont été positionnés et qu’un gabarit a été créé, le menu permettant d’accéder au résultat de la fusion, « Fusion / ResultatFusion ». Si l’un de ces éléments n’a pas été créé, alors le menu ne sera pas accessible car il sera impossible de réaliser la fusion.

Lorsque vous allez lancer l’interface, vous n’obtiendrez pas directement le résultat de la fusion, vous allez devoir réaliser quelques manipulations :

- Cochez les deux cases « Echographie » et « Photo de la peau » dans la partie « Gestion de la transparence » à droite. Ensuite, vous pouvez choisir laquelle des deux images (entre le scan C et la photo de la peau) pour définir celle qui sera au premier plan.

- Modifiez la transparence des images pour obtenir un résultat avec la meilleure qualité possible.

- Modifiez le positionnement de Z pour afficher le scan C que vous souhaitez.

- Si vous souhaitez vous repérer sur la photo, vous pouvez utiliser la photo sur la gauche et afficher ou non les points repères que vous avez placés et ceux qui ont été recalés

VII- Sauvegarder et importer un projet

Une fois que le résultat de la fusion est accessible, alors le menu « Fichier / Sauvegarder projet » est également accessible et il vous est alors possible de sauvegarder votre travail pour le réutiliser plus tard. Le fichier de sauvegarde créé est un fichier texte, et c'est à partir de ce même fichier que vous pourrez importer le projet dans l'application. Tout votre travail sera alors chargé en mémoire et vous retrouverez l'application comme vous l'aviez quitté après la sauvegarde.

Recalage 2D-3D en dermatologie

Département Informatique
5e année
2011-2012

Rapport de projet de fin d'études

Résumé : Ce rapport a été écrit à la suite d'un projet de fin d'études à l'école d'ingénieurs Polytech'Tours, département informatique. Principalement basé sur la modélisation et le développement d'une application permettant aux praticiens en dermatologie de mieux visualiser une lésion ou une tumeur sous cutanée, ce rapport reprend les différentes étapes qui ont permis de réaliser l'application finale.

Mots clefs : Dermatologie, recalage, échographie 3D, fusion.

Abstract : This report has been written for a last year project in the engineering school Polytech'Tours, IT section. The main part of the report deals with the conception and the development of a medical software (dermatology) which allow the practitioner to have a better subcutaneous view of a tumor. Every single step of the project are detailed in this report.

Keywords : Dermatology, repositioning, 3D scan.

Etudiant :

Alexandre NEVEU
<alexandre.neveu@etu.univ-tours.fr>

Encadrants :

Pascal MAKRIS
<pascal.makris@univ-tours.fr>
Jean-Yves RAMEL
<jean-yves.ramel@univ-tours.fr>
Baudouin MARTIN
<baudouin.martin@univ-tours.fr>